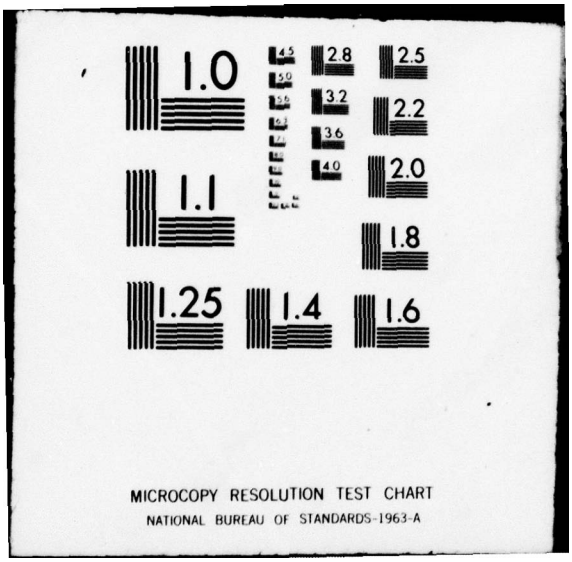
AD-A063 430 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON D C F/G 9/2
NMCS INFORMATION PROCESSING SYSTEM 360 FORMATTED FILE SYSTEM (N--ETC(U)
SEP 78 C K HILL
UNCLASSIFIED CCTC-CSM-UM-15-78-VOL-5 SBIE-AD-E100 130 NL

1 OF 2
AD
A063430


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(12) NW

COMPUTER SYSTEM MANUAL
CSM UM 15-78
VOLUME V
1 SEPTEMBER 1978

C C T C

COMMAND & CONTROL TECHNICAL CENTER

LEVEL III

NMCS INFORMATION PROCESSING SYSTEM 360 FORMATTED FILE SYSTEM (NIPS 360 FFS)

VOLUME V
OUTPUT PROCESSOR (OP)

USERS MANUAL

DEFENSE COMMUNICATIONS AGENCY

(14) CCTC-CSM-UM-15-78-VOL-5

COMMAND AND CONTROL TECHNICAL CENTER

Computer System Manual Number CSM UM 15-78

(11) 1 Sept[redacted]978

(6)

NMCS INFORMATION PROCESSING SYSTEM 360 FORMATTED FILE SYSTEM (NIPS 360 FFS).

User's Manual.

Volume V. Output Processor (OP).

(12) 156 p.

(9) Computer system manual;

SUBMITTED BY:

CRAIG K. HILL
Captain USA
CCTC Project Officer

(18) SBIE

(19) AD-E100 130

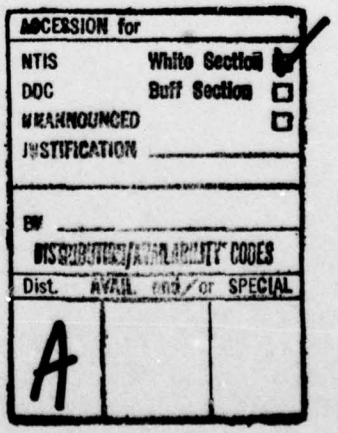
APPROVED BY:

FREDERIC A. GRAF, JR.
Captain U.S. Navy
Deputy Director
NMCS ADP

(10) Craig K. Hill

ACKNOWLEDGMENT

This manual was prepared under the direction of the Chief for Programming with general technical support provided by the International Business Machines Corporation under contracts DCA 100-67-C-0062, DCA 100-69-C-0029, DCA 100-70-C-0031, DCA 100-70-C-080, DCA 100-71-C-0047, and DCA 100-77-C-0065.



CONTENTS

Section Page

ILLUSTRATIONS

TABLES

ABSTRACT

This volume describes the Output Processor (OP) component of the NIPS 360 FFS. It presents the Output Processor's capability, the type of output produced, and the methods employed in producing this output. It describes the control cards required, a quick reference summary of all cards, and a summary of differences between 1410 NIPS and NIPS 360 FFS concepts of the Output Processor.

This document supersedes CSM UM 15-74, Volume V.

CSM UM 15-78, Volume V is part of the following additional 360 FFS documentation:

CSM UM 15-78	Vol I	- Introduction to File Concepts
	Vol II	- File Structuring (FS)
	Vol III	- File Maintenance (FM)
	Vol IV	- Retrieval and Sort Processor (RASP)
	Vol VI	- Terminal Processing (TP)
	Vol VII	- Utility Support (UT)
	Vol VIII	- Job Preparation Manual
	Vol IX	- Error Codes
TR 54-78		- Installation of NIPS 360 FFS
CSM GD 15-78		- General Description

OUTPUT PROCESSOR (OP)

Section 1

INTRODUCTION

This volume describes the Output Processor (OP), its various capabilities, and the language required to use it. OP is a valuable aid for data presentation.

The reader will understand terms used in this document more readily if he has a general understanding of NIPS 360 FFS file concepts.

This volume is divided into the following sections.

OP Capabilities - General description of the processor's capability, the type of output produced, and the method used in producing this output.

Control Card Formats - Control cards required to specify the format and publication of reports on a variety of output media.

Report Specifications Summary - Quick-reference summarization of all cards used in Report Instruction Table (RIT) structuring.

Run Deck Formats - Typical sequences of Output Processor control cards.

Supporting Features - Those features which complement the Output Processor or are required for its operation.

Summary of 1410-S/360 Changes - Significant differences which NIPS 1410 FFS users will encounter in using NIPS 360 FFS.

Section 2

OP CAPABILITIES

The Output Processor is designed to provide the user with hard copy reports, printed listings, punched cards, or magnetic tape of the data file or retrieval answer files. The system enables the user to see the information stored and to use the various report structuring options to control the output format and alter the data content by editing, subroutine conversion, or arithmetic operations.

The user communicates his requirements to OP through report-structuring cards. OP translates the cards and generates machine-language instructions to perform the output logic. These instructions constitute a RIT and may be stored on a temporary data set for use on a one-run basis or on a library data set for repeated usage.

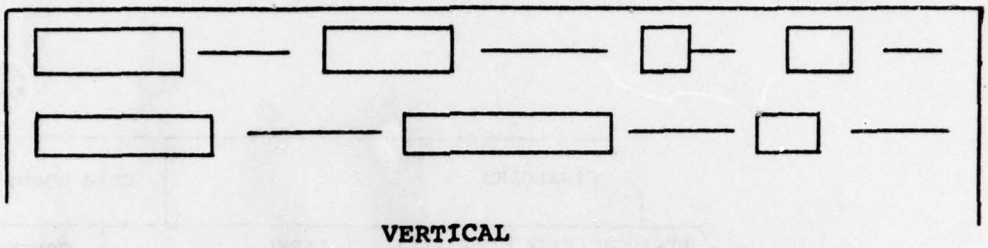
2.1 Input

Input to the Output Processor component may be a QRT/QDF produced by the RASP component, or in Source Direct mode, a NIPS data file. When the data file is the input to OP, the data file may be a single sequential file, concatenated segments of a sequential file, an indexed sequential file, or S/370 virtual storage (VSAM) file.

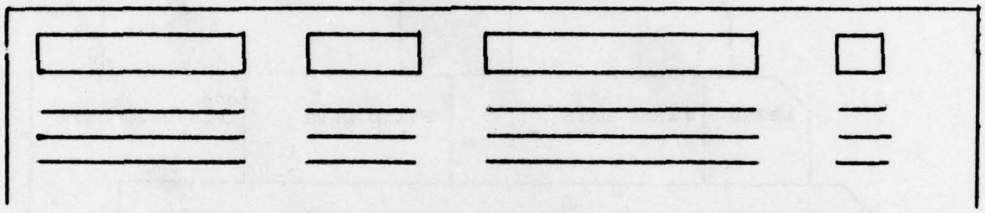
2.2 Output

Figures 1, 2, and 3 illustrate the results which are obtainable from the Output Processor.

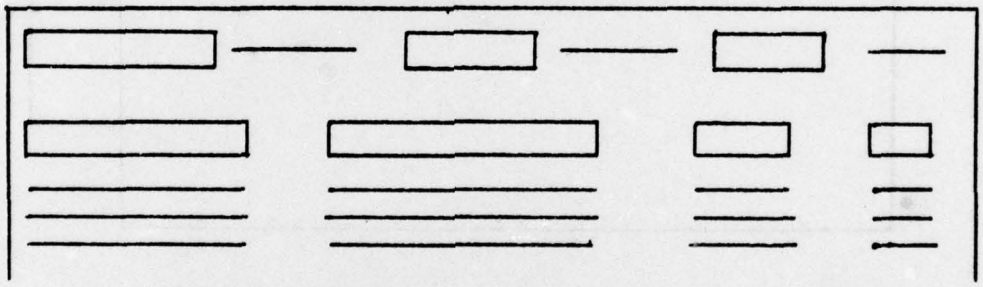
HORIZONTAL



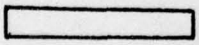
VERTICAL



COMBINATION



NOTE: Labels may be inserted on any line

 LABELS

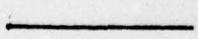
 DATA

Figure 1. Data and Label Printout

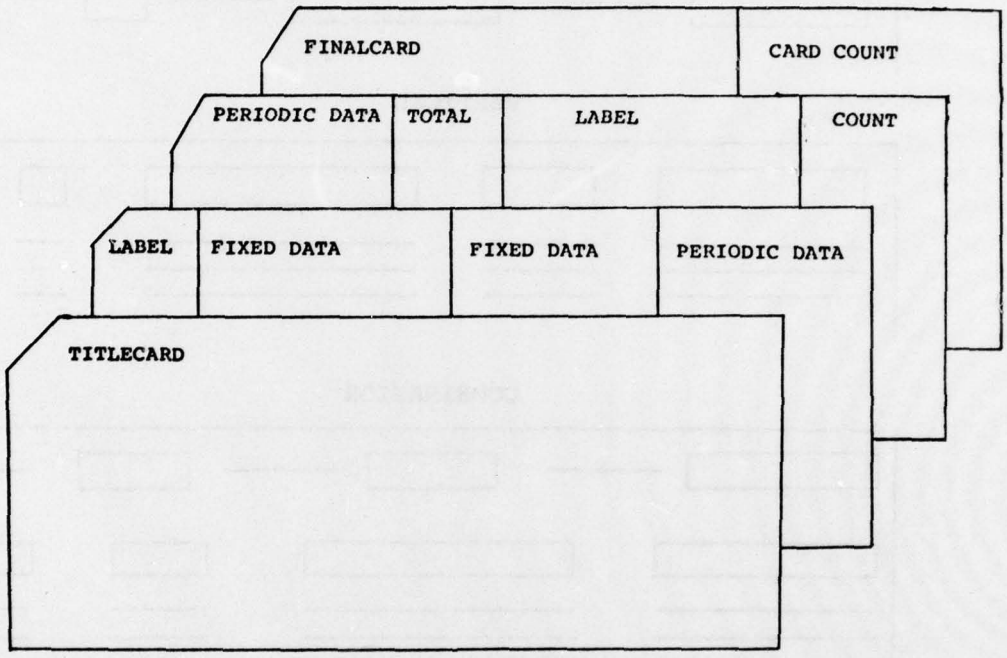


Figure 2. Punched Output

OUTPUT PROCESSOR (OP)

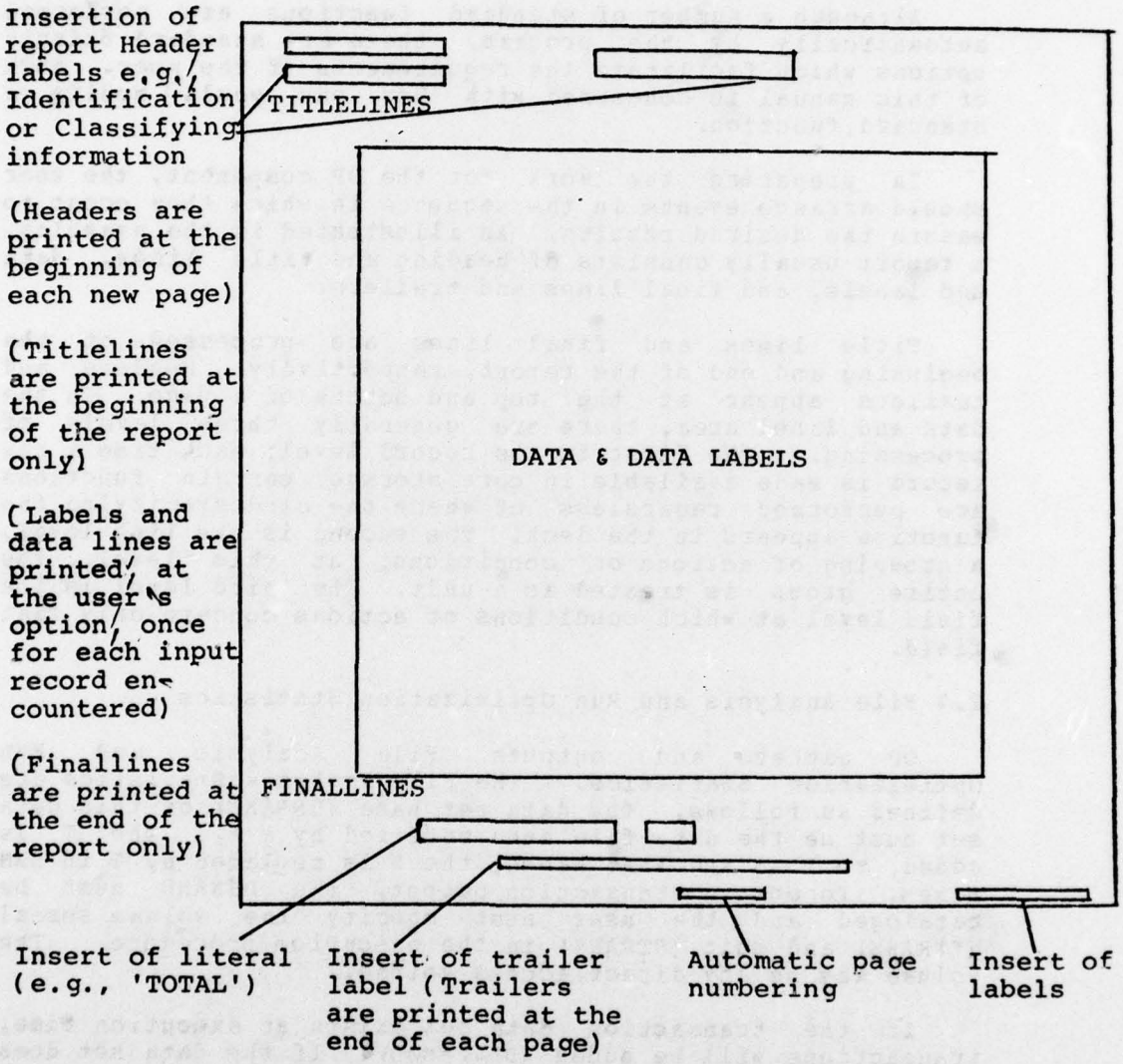


Figure 3. Printed Output

2.3 Standard Functions

Although a number of standard functions are performed automatically by the program, there are standard default options which facilitate the requirements of the user. Much of this manual is concerned with how one would modify a standard function.

In preparing the work for the OP component, the user should arrange events in the sequence in which they occur to ensure the desired results. As illustrated in the examples, a report usually consists of heading and title lines, data and labels, and final lines and trailers.

Title lines and final lines are processed at the beginning and end of the report, respectively. Headers and trailers appear at the top and bottom of a page. In the data and label area, there are generally three levels of processing. The first is the record level; each time a new record is made available in core storage, certain functions are performed regardless of where the card specifying the function appears in the deck. The second is the line level, a grouping of actions or conditions; at this level, the entire group is treated as a unit. The third level is the field level at which conditions or actions concern only that field.

2.4 File Analysis and Run Optimization Statistics

OP gathers and outputs File Analysis and Run Optimization Statistics. The File Analysis Statistics are defined as follows. The data set name (DSNAME) of this data set must be the data file name suffixed by a T. The T is added to ISAM and VSAM names; the S is replaced by T in SAM names. To obtain transaction output, the DSNAME must be cataloged and the user must specify the volume serial (VTRANS) and unit (UTRANS) in the execution procedure. The volume may be any direct access volume.

If the transaction data set exists at execution time, transactions will be added (DISP=MOD). If the data set does not exist, a 5-track data set will be dynamically allocated. The user may change the allocation value by overriding the TRANST DD card space parameter. Transactions are written as fixed length, unblocked, 50-byte records. The format

(fixed) and length (50) cannot be changed but the user may change the blocking factor by specifying a DCB BLKSIZE in the TRANST DD card which is a multiple of 50.

If the user specifies a DSNAME (TRANS) in the TRANS DD card, he must supply all parameters required to process the data set. These parameters must conform to the requirements defined above.

The statistics gathered will be in the format of transaction records suitable for input to an FM run to update a file. The information consists of the data file name, component name, source module name, count of executions of the source, and the date the source module was executed.

The Run Optimization Statistics are initiated through parameters entered in the PARM field on the EXEC card. The breakdown of the statistics details the amount of core used for user subroutines and tables, RIT, process block, I/O buffers, and access methods. It will also include the number of BLDL entries allocated and used, and the number of entries required for each subroutine, table, and the RIT to reside in core. The amount of core required for each of them to reside in core is output. If any are rolled, this information is output with the number of times rolled and the cause of the rolling.

Since OP uses all of the available core for a processing block, the value in the process block used field will be a minimum size necessary to hold the largest data record, fixed set, and all periodic subsets.

The user is able to enter override parameters for the number of BLDL entries to allocate and the size of the processing block desired for the storage of data records during OP execution.

The parameters that may be entered in the PARM field on the EXEC card are as follows:

ROS - Indicates the Run Optimization Statistics are to be gathered and output.

NOROS - Omit Run Optimization Statistics. If none of the following parameters are used, this should be omitted as it is the default.

The parameters that may be used to tailor the core allocation are as follows:

TCP=nK - The number (n) of 1000 (K) bytes requested for the process block.

TCB=n - The number (n) of BLDL entries to allocate.

TCS - Use the statistics record on the ISAM data file to compute process blocksize. This parameter cannot be used with the TCP parameter.

For a more detailed description of the capability, see Volume I, Introduction to File Concepts.

2.5 Limitations

The user of the Output Processor should be aware of the following system limits:

40	RITs structured in one OP run.
10	Data files in one RIT.
42	Sets referenced in one RIT.
30	Subroutines specified in one RIT.
550	Named literals defined in one RIT.
52	Maximum literal size (named or unnamed).
8192	Bytes for a compiled individual level 1 statement.
5	Retriever generated work area values per set; each is one fullword (four bytes) in length.

There is no limit on the number of the following items which may be specified:

OUTPUT PROCESSOR (OP)

Data field names

Data lines

Label lines

Header lines

Trailer lines

Record(s)

Card(s)

Control cards

Section 3

CONTROL CARD FORMATS

This section describes the control card requirements to specify the format and publication of reports. The cards required to supervise an Output Processor run are specified first. General requirements for structuring a RIT are given, followed by detailed information for specifying reports on three output media--printer, card punch, and magnetic tape.

3.1 Output Supervisor Control Cards

The control cards designed for use with the output supervisor are described in this subsection. All control cards, except where noted, are in free format with the card type beginning in column 1; the other card entries follow this card type with one or more spaces between each entry or word. Entries may be placed in the card through column 71. Any punch in column 72 indicates that there is a continuation card.

If required, sequence numbers for the control cards are punched in columns 73 through 80.

Before describing the individual control cards the general order of a run deck will be described. The Output Processor executes previously structured RITs using as input either a data file or an answer file generated by a retrieval. Thus a run deck may contain RIT specification decks or appropriate control cards to execute a RIT. Consider a run deck as having two parts separated by a SOURCE card. The order would be:

OUTPUT PROCESSOR (OP)

RIT specification decks

SOURCE card

Publish control cards

3.1.1 CREATE Card (RITID, STORE, DEBUG, BOOL)

Each RIT to be structured must be preceded by a CREATE card. The CREATE card identifies the RIT by assigning an RITNAME which must conform to the NIPS naming conventions (section 2.6.3 Volume I, Introduction to File Concepts) and directs the disposition of the generated program. Permanent or standard RITs are added to or replace existing RITs on a permanent library and remain there until deleted by the user. Temporary RITs are placed on a temporary library and are released at the end of the current run.

This card may also be used to specify two special operators, DEBUG, which indicates that the structure deck is to be checked for errors only, and a BOOL option which indicates that Boolean logic is to be used with conditions.

The CREATE card has a positional keyword format. The allowable keywords are as follows:

CREATE - Identifies the card and starts in column 1.

RITID - Is the keyword for RIT identifications.

STORE - Is the keyword for disposition. Disposition can have the values PERM-OLD to replace a permanent RIT, PERM-NEW to add a permanent RIT and TEMP for a temporary RIT.

DEBUG - Designates error check only.
This option provides for validation of a run deck without building a RIT. The STORE parameter should not be used if DEBUG is specified.

BOOL - Used as the last parameter to indicate Boolean logic used.

Note: If the keyword RITID= is not first, the value specified is assumed to be the name of a RIT to be

structured, and the following options are taken: storage of the RIT on the permanent library as a change and BOOLEAN logic. When the keyword RITID is specified, the second entry must be STORE or DEBUG; if any other value is specified, the DEBUG option is assumed and the RIT will only be checked for specification errors.

Example:

a. CREATE RITAR
RITAR will be structured and stored on the permanent library as a change. The Boolean logic option is assumed.

b. CREATE RITID=RITNAME STORE=PERM-OLD
Structure the RIT RITNAME. The store option is used to replace a permanent RIT on a library.

c. CREATE RITID=RITNAME STORE=PERM-NEW
Same as example b except that the newly structured RIT is added to the permanent library.

d. CREATE RITID=RITNAME STORE=TEMP
The RIT will be added to the temporary library and deleted on completion of the step.

e. CREATE RITID=RITNAME DEBUG
A debug pass will be made through the edit and translator to check for specification errors. No RIT will be structured or stored.

f. CREATE RITIT=RITNAME STORE=PERM-NEW BOOL
Specifies Boolean logic option.

3.1.2 SOURCE Card (DIRECT, RETRIEVAL)

The last RIT specification deck (if any) is followed by a SOURCE card, if reports are to be published during the same job. There are two sources of input data to OP.

SOURCE DIRECT - Used when data is to be read directly from the data file without having previously been qualified by retrieval. It must be followed by at least one PUBLISH card.

Example:

To run directly against a data file and to publish the report produced by RIT ZILCH.

```
SOURCE DIRECT
PUBLISH RITID=ZILCH
```

SOURCE RETRIEVAL - Used when data is read from a file qualified and produced by the Retrieval program. Operating in a batch query environment, one can expect more than one set of answers on the source data set. If all answer sets are to be published using the RITs specified at retrieval time and not requiring any additional input parameters, no other control cards are required. This is referred to as the "nonselect" mode.

Example:

```
SOURCE RETRIEVAL
```

This card alone would indicate that all answer sets on the input data set were to be published by the retrieval designated RITs that are on the Permanent Library. If only designated answer sets are to be published, or additional parameters are to be provided, the SOURCE card will be followed by at least one PUBLISH card per report published. This is referred to as the "select" mode.

Example:

```
SOURCE RETRIEVAL

PUBLISH ANSID=17....
```

Answer set number 17 would be processed.

3.1.3 PUBLISH Card (RITID, SPECIAL, ANSID, COVERPAGE, PAGENO, BODYLINES, COPIES, CLASS, PARAM, DEBUG, DATE)

The PUBLISH card is used to provide the OP Supervisor with the RIT name, the answer set ID, and a cover page code.

The PUBLISH card is free format, keyword in form. Columns 73-74 must be blank. All parameters required by the Output Processor can be entered using this card.

OUTPUT PROCESSOR (OP)

Keyword Example	Definition
RITID=RITNAME	Identifies name of RIT stored on Permanent Library.
SPECIAL=RITNAME	Identifies name of RIT stored on Temporary Library.
ANSID=NN	Identifies a set of answers to be published. NN is the 1- to 4-digit query number used at retrieval time. Note that for multiple executions of the same query, the retrieval component will suffix the query number with an alpha character for the *second* and subsequent executions. The suffixes begin with the letter "A" and must be included as part of the ANSID when applicable.
ANSID=NN/RRR	Identifies a set of answers to be published. NN is the query number as defined above and RRR is the secondary (RIT) ID used at retrieval time.
COVERPAGE=CDW	Designates the cover page code to be used for the report. (Optional.)
PAGENO=NN	Designates a starting page number; defaults to 1. (Optional.)
BODYLINES=NN	Allows number of body lines per page to be

	changed at run time; defaults to 50. (Optional.)
COPIES=NN	Allows number of copies to be specified; defaults to 1, (Optional.)
CLASS=UNCLASSIFIED	Classification of output. Note that if there are embedded blanks the classification must be enclosed in (quote) signs.
PARAM='PARAM EXAMPLE'	Parameter allowing the user to introduce up to 60 bytes of information to the RIT at RIT execution time. Same rule for embedded blanks.
DEBUG=NN	Used at execution time to allow the user to designate the number of data blocks to be passed against the RIT. The maximum number specified is 32,767. When N is exhausted, an end-of-file or end-of-answer set is simulated. When data interrupts occur and a dump is required, the DEBUG parameter can be used. If there is a debug count, the first data interrupt will cause a SYSABEND dump. When using this option, make sure the debug count is high enough to get to the record causing the data interrupt.

OUTPUT PROCESSOR (OP)

DATE=DDMMMYYHHMM — Allows user to specify up to an 11-character date. Same rule for embedded blanks.

Example:

```
PUBLISH ANSID=1 RITID=RITEXMP COVERPAGE=CDW                    X
PAGENO=101 BODYLINES=40 COPIES=2                               X
CLASS='VERY UNCLASSIFIED' DATE='15FEB68'                       X
PARAM='THIS IS AN EXAMPLE OF THE PUBLISH CARD'                 X
DEBUG=10
```

Several of the PUBLISH card parameters warrant further explanation.

ANSID may specify a query number or a query number and a secondary ID. If the retrieval produced several subanswer sets and ANSID=NN is used, all subanswer sets will be produced. If a particular subanswer set is required the form ANSID=NN/RRR must be used.

Execution of a temporary RIT can be specified only via this form of PUBLISH card. This includes retrieval-designated RITs.

Care should be used in specifying the query number for publishing multiple executions of the same query by the retrieval component. The second and succeeding query executions result in the query number being suffixed with an alpha character. Suffixing begins with "A" and progresses sequentially for each execution of the same query. For example, if query number "22" was executed three times (with different replacement values using the RASP skeleton capability), the first answer set would be specified as ANSID=22, the second as ANSID=22A and the third as ANSID=22B. The specification of the subanswer set (RIT ID) remains unchanged.

Publication of an answer set can be requested any number of times in the same run. The PUBLISH control cards can be in any order. Answers are published in answer file order.

Example:

```
PUBLISH     ANSID=1     RITID=RITNAME

PUBLISH     ANSID=1     SPECIAL=ZILCH
```

The CLASS parameter is compared against the file classification and if not equal, or if the classification parameter is not specified, a warning page will be printed along with a console diagnostic to the operator.

The DEBUG parameter is a useful tool in debugging a RIT. The analyst can "run" against a real data file or answer set, and limit the amount of output to a specified number of blocks until checked out.

COVERPAGE is a series of codes providing a special cover page for the report containing classification downgrade and warning information in the format COVERPAGE = CDWS.

In compliance with the Executive Order 11652, 8 March 1972 which establishes current policies governing classification, downgrading and declassification of official information, the following coverpage features are available for the user's requirements.

The codes and their results on the cover page are as follows:

The classification code is represented by 'C' in the example.

C - Code Data

Code	Data
U	Unclassified
C	Confidential
S	Secret
T	Top Secret
J	Top Secret - Crypto
I	Top Secret - Special Intelligence
H	Top Secret - Category 10
G	Top Secret - Category 7
F	Top Secret - SIOP
E	Top Secret - SIOP - ESI

Note: The following paragraphs are printed with Code F--Top Secret - SIOP.

Special handling required--not releasable to foreign nationals or their representatives.

This document contains information affecting the National Defense of the United States within the meaning of the Espionage laws, Title 18, U.S.C., Sections 793 and 794. The transmission or the revelation of its contents in any manner to an unauthorized person is prohibited by law.

The downgrade code is represented by 'D' in the example. The absence of a valid code causes the program to bypass the printing of the cover page. This does not cause the OP run to be terminated. Valid codes and their corresponding results are as follows:

OUTPUT PROCESSOR (OP)

D - Code Data

---- ----

a. DOWNGRADE TO:_____
SECRET ON_____
CONFIDENTIAL ON_____
DECLASSIFY ON_____

b. CLASSIFIED BY:_____
EXEMPT FROM GENERAL DECLASSIFICATION SCHEDULE OF EXECUTIVE ORDER 11652 EXEMPTION
DECLASSIFY ON_____

c. CLASSIFIED BY_____
SUBJECT TO GENERAL DECLASSIFICATION SCHEDULE OF EXECUTIVE ORDER 11652 ATUOMATICALLY DOWNGRADED AT TWO YEAR INTERVALS DECLASSIFY ON 31 DECEMBER_____

The warning code is represented by 'W' in the example. This code is optional and is used to designate that a special warning should be printed on the cover page. The warning codes and their corresponding results are as follows:

W - Code Data

---- ----

A Controlled dissemination

B Formerly Restricted Data, Section 144B, Atomic Energy Act, 1954

C Restricted Data, Atomic Energy Act, 1954

D Special Handling Required, not releasable to foreign nationals or their representatives

E Reproduction of document in whole or part prohibited except with permission of issuing office or higher authority

F Special Handling Required, not releasable to foreign nationals or their representatives, reproduction of document in whole or part prohibited except with permission of issuing office or higher authority

The special instruction code is represented by 'S' in the example. It is used to designate one of the following instructions:

X NATIONAL SECURITY INFORMATION
UNAUTHORIZED DISCLOSURE SUBJECT TO
CRIMINAL SANCTION

(Note: This instruction is for classified information which is furnished to persons outside the Executive Branch)

W WARNING NOTICE - SENSITIVE INTELLIGENCE SOURCES
AND METHODS INVOLVED

Note: For all codes the letter "N" (for none) may be used in any position to omit blocks of options.

3.2 Report Structuring Control Cards

The following is a summary of OP control cards used for structuring a RIT. A FILE card identifies the file for the specifications to follow. A FORMAT card describes the format of tape, card, or printed listings and the limits of each if required (e.g., blocksize, number of positions per line for the specifications to follow).

Either a FILE card or FORMAT card must appear first. This will allow multiple files by format, or multiple formats by file.

Omit - Followed by conditional logic; used to omit presentation of a record.

Stop - Followed by conditional logic; used to stop the report.

Header - Describes information that will appear at the top of each page of printed output.

Titleline - Describes information that will be printed at the outset of the report (albeit after header's information) only.

Overflow - Information that will be printed (following header information) if the first line (LINE1) of the RIT is not the line to be printed after the printer ejects to a new page. (See figure 4 for an example.)

HEADER

Title Page 1

Line1
Line2
Line3
Line4

TRAILER
HEADER

Overflow Page 2

Line5
Line1
Line2
Line3

Figure 4. Overflow Line

Line - The primary method for specifying data information on the printed page. In a RIT calling for multiple sets of line specifications, the lines will be printed. Periodic information will be repeated until all such information is printed, going through a complete cycle for each record.

Label - Associated or supplementary information attached to a line; an associated label will be printed only if its line is printed.

Finalline - Information to be printed only at the termination of the report.

Trailer - Information that will appear at the bottom of each page of printed output.

The following paragraphs discuss in detail the report specifications necessary to generate a normal report format. The discussion of the report specifications introduce examples to illustrate the specifications. Specifications for punch card and magnetic tape format are shown in sections 3.4 and 3.5 respectively.

3.2.1 Specification of the Data File

The example used here is a single file report and the first specification should be the file mnemonic to which the report pertains. This is true whether more than one format (print, punch, or tape) is to be specified or if the report is to be printed only, as in this example. Multifile reports must have the file mnemonic specified for each set of specifications dealing with that file. Thus, for this example, the file mnemonic would be:

```
FILE TEST360
```

The file mnemonic specified on the FILE statement must correspond to the unqualified data set name or the last segment of a qualified data set name designated in the JCL.

3.2.2 Formatted Reports

The Output Processor can provide formatted reports printed on all sizes of printer paper, punched in cards, or written on magnetic tape. To indicate the type of output desired and to specify the exact format, report specification cards are used.

3.2.3 Report Instruction Table (RIT)

Report specification cards are the input to the Report Structuring program of the Output Processor. The cards are analyzed and the specifications are built into a RIT and any errors are printed for review by the user. Some errors are considered as diagnostics only and will allow the

compilation of the RIT to be completed while others will cause the run to be terminated before the assembly phase. In some instances assumptions will be made which will allow the RIT to be structured; in any event, all the errors and diagnostics are listed.

3.2.4 Report Specification Cards - General

Output media are indicated on format cards along with specifications for overall format control. The file mnemonic to which the specifications pertain is indicated on the FILE card.

Under normal conditions the next item for consideration is the layout of the lines, cards, or tape records, the specification of the positions of the data, and any data manipulation or other output requirements. Therefore, any header lines are defined (HEADER card) followed by the data to go on each line, card, or tape record (LINE, CARD, or RECORD cards, respectively). Such operations as edits, conversion tables, output subroutines, calculations, or logical output requirements are also specified on these cards. Label lines, which are dependent upon the output of data lines, may be specified on LABEL cards. Finally, the trailer lines are defined (TRAILER card).

The normal spacing between lines can be indicated on the FORMAT card; other spacing can be indicated by SPACE cards. The format of the report can also be controlled with the carriage control tape by the use of the SKIP card, and an ejection to a new page by the EJECT card.

The above card types are discussed in detail in the following sections along with a discussion of more complex reports and additional specifications necessary to handle these reports.

3.2.5 Card Layout

All specification cards, except where noted, are in free format with the card type as the first entry on the card. The succeeding entries on each card are separated by one or more blanks. The entries may be placed through column 71 of the card and in certain cases it is necessary to continue the entries on the next card. If this is required, each

card which has specifications continuing on a following card must have a punch in column 72 and the continuing card must have the card type as the first entry (this grouping is limited to three cards); otherwise, column 72 must be blank. Only three continuation cards may be used providing a total of four cards for the compute statement.

Columns 73-80 are provided for identification and card sequencing. It is recommended that columns 73-76 be used for deck identification. Columns 77-80 are for the card sequence number. A sequence check is performed on columns 77-80 unless these columns of the first card contain blanks. However, any errors detected will not affect the analysis of the cards. If the sequence number is blank or if a sequence error is found, the Output program will insert a sequence number ending with S and all further references to the card will be with this number.

All output positions specified for data fields or literals must be low-order (rightmost) positions.

Comment cards (indicated by an asterisk in column 1) may be inserted anywhere in the deck.

3.2.6 Control of Page Format

The format of a page is specified through the use of a FORMAT card. For a printed format, the card will always contain the word PRINT following the word FORMAT.

```
FORMAT PRINT
```

Body lines are the number of lines per page exclusive of header and trailer lines. After determining the page layout with the header and trailer lines and their associated spacing, the number of body lines desired per page (the number of lines, including spacing, between the last header line and the first trailer line) may be specified. This is indicated by including the term LINES on the FORMAT card followed by the number of body lines required. If the LINES entry is omitted, the standard mode of 50 body lines will be assumed.

```
FORMAT PRINT LINES 55
```

Normal Spacing Between Lines

The option is provided to specify an overall spacing between lines for the report. The maximum spacing that can be specified is four spaces. The spacing indicated will be used by the output program unless a SPACE card is used to override this spacing.

The spacing factor is indicated by including the term SPACE on the FORMAT card followed by the number of spaces required.

```
FORMAT PRINT  SPACE 2
```

The above example specifies double spacing and becomes the normal spacing. If this entry is omitted, single spacing will be assumed.

Paper Width

This option is included to provide additional error checking for the print output positions specified. The paper size is specified by including the term SIZE on the FORMAT card followed by the number of print positions for the width of the paper used for the report.

```
FORMAT PRINT  SIZE 105
```

The above example specifies the width of the report to be 105 print positions. If this entry is omitted, the standard 132-position paper will be assumed.

The above specifications can be entered on the format card in any order after the word PRINT. For example :

```
FORMAT PRINT  SPACE 2  SIZE 105  LINES 55
```

The above example combines all three options.

3.2.7 Specification of System Labels (OPDATE, PAGENO, CLASSIF, BODYLINES, PARAM, PSCTn, WORKn)

The Output Processor provides the ability to specify certain parameters when the report is produced. These

parameters are specified through the use of system labels at the time the RIT is compiled, and with their respective output control cards at execution time.

OPDATE

This term defines a field which holds a specified date which is to be used for this report. It is an 11-character field that can be entered on a date card at the time a report is produced. The Output program will use the date as specified on the control card. If the OPDATE card is omitted at execution time, the Output program will use the system date, in the form 'DD MMM YYYY'.

PAGENO

This term defines a 6-digit field which holds the current page number.

CLASSIF

This term defines a 32-character field which holds the classification for this report. If the CLASSIF card was omitted, the field "CLASSIF" will be blank.

BODYLINES

This term identifies a 2-digit counter that holds the current number of body lines remaining on a page.

PARAM

This term defines a 60-character area into which data can be introduced to the RIT at execution time. The data can be used for control purposes or data presentation.

PSCTn

This term defines a field which holds a count of subsets loaded into the processing block. A RIT which references PSCTn will reference this count. Note that only subsets used by the RIT are loaded; i.e., if the RIT uses only subsets flagged in a retrieval run, nonflagged subsets will not be loaded. The set number is designated by n. The PSCT

field may be referenced only in MOVE expressions, on print positions, or in conditionals.

WORKn

This special term must be used when referencing data values (work areas) passed to the Output Processor by the RASP. There are five work areas (WORK1 through WORK5). Each area is one fullword (four bytes) in length. The work areas are set associated and are processed, using the same rules and restrictions as file data fields. The user must specify the set, a name for each work area, the output length and the data mode. Specification of literals or work areas in the general case, and this special case, is made in a DEFINE statement as described in paragraph 3.3.4.

3.2.8 Conditional Statements (IF)

Many of the specifications to be described may be conditional and may be expressed similar to the terms in retrieval language. Conditional statements are of the general form:

... If, parameter, condition, parameter, AND/OR, parameter, condition, etc.

These conditional expressions are used to condition the printing of fields, lines, or results of computations. It should be noted that only conditions of level 2 will condition the writing of a line, card, or record. Conditions at levels 4 or 5 on a line, card, or record statement condition the moving of data or the action specified on the statement but not the writing of the output data.

Generally, the parameters may be any field or group name except that both parameters must not be periodic fields or groups from different sets. The last parameter may also be a previously defined literal, an alphabetic literal ('literal'), a numeric literal, or binary literal. The first parameter must be a previously defined literal or field.

Note: Crossing set boundaries between action and condition creates an illogical statement. A fixed set conditioned on

any periodic is permitted. A periodic set may be conditioned on the same periodic. A periodic set conditioned on a different periodic set is not permitted. This will generate an error condition. An example of illegally crossing set boundaries is: LINE5 20 MEQPT IF PLAN EQ 2222 (where MEQPT and PLAN are in different periodic sets).

The conditions which may be used are similar to those used in retrieval and are listed as follows:

Negation	Greater Than	Less Than	Equal To	Change From Previous Value
NOT	GREATER	LESS	EQUAL	CHANGE
NE (a special case of not equal)	LATER	EARLIER	EQUALS	CHANGES
	AFTER	BEFORE	EQUALING	CH
	GT	LT	EQ	

Note: See table 2 of section 4 for a complete list.

The CHANGE operator provides OP with the capability to perform specified actions when values change from a previous state. The previous value, used as the basis for CHANGE logic, is updated each time the term containing IF CHANGE is processed. It is not updated when the term has been bypassed in the logical processing of all conditions/terms in the RIT. Bypassing is possible when multiple change terms are connected by an OR, and the user should understand that CHANGE logic may not be based on the value in the immediately preceeding record. If this is not acceptable, flags may be substituted, for example:

```
MOVE 'X' TO FLAG1 IF SERV CHANGES
     .
     .
     .
LINE14 IF FLAG1 EQ 'X'
     .
     .
     .
```

```
LINE14 MOVE ' ' TO FLAG1
```

In addition to the above conditions, OP makes use of two other special purpose conditional operators. These are CONTAINS and ABSENT.

The CONTAINS operator provides OP with a text scan capability that recognizes field values wherever they occur within a fixed or variable length field or variable set. A scan of a specified field will occur in a shifting comparison technique to determine whether the condition is satisfied. In this manner, the user may access variable fields, variable sets, and data that is in any portion of a fixed-length alpha field. A conditional statement using CONTAINS is written in the following manner:

```
... IF fieldname CONTAINS parameter.
```

When using the ABSENT operator, a condition is satisfied when there are no subsets in the referenced set. The referenced set may be either a periodic or variable set. A conditional statement using the ABSENT operator is written in the following manner:

```
... IF fieldname ABSENT.
```

Certain other (noise) words may be included to improve readability.

Noise Words

IS

THAN

THE

TO

The conditional statement will be tested using the S/360 logical collating sequence unless both parameters are numeric. In using a conditional statement, standard rules of truncation and padding are used. If both parameters are numeric, an algebraic comparison will be used.

OUTPUT PROCESSOR (OP)

Two methods for logically processing conditions are provided. They are referred to as "standard" and "Boolean."

The Output program examines the "standard" conditional statement in the following manner:

a. If an AND or IF term is satisfied, the following OR terms are not processed. The next term to be checked is another AND term.

b. If an AND or IF term is not satisfied, succeeding OR terms are examined until one is satisfied. The remaining OR terms are not checked, and the next term to be checked is the next AND term.

c. If an AND or IF term is not satisfied, and no OR term is satisfied prior to another AND term or prior to the end of the expression, the condition fails.

d. If the end of the logic condition is reached and the preceding AND/OR group is satisfied, the logic condition is satisfied.

The following examples are given to illustrate the above:

Expression	Interpretation
IF A AND B	Both A and B must be true.
IF A OR B	Either A or B must be true.
IF A OR B AND C	C must be true. Either A or B must also be true.
IF A AND B OR C AND D	A must be true, D must be true, and either B or C must be true.

"Boolean" logic processing differs from the standard RIT conditional logic processing. The ANDs tend to join and ORs tend to separate. The following logical truth tables illustrate the processing of the Boolean logic:

Expression	Results
IF A AND B	Both A and B must be true.
IF A OR B	Either A or B must be true.
IF A AND B OR C	Both A and B must be true, or C must be true.

The sequence of processing for a condition statement is as follows:

a. Starting with the first clause (AND), each AND clause is processed until: (1) the end of the statement; (2) a false condition is found; or (3) an OR is reached.

b. If the AND (string) is terminated because of a false condition, processing resumes at the next OR. If there is not another OR before the end of statement, the condition statement is false.

c. If the AND (string) is terminated because of reaching the end of statement or an OR is reached, that statement is true.

The difference between the standard conditional logic and Boolean logic is illustrated by the examples below.

Standard Conditional Logic

Expression	Interpretation
IF A AND B OR C	A must be true; either B or C must be true.
IF B AND A OR C	B must be true; either A or C must be true.

Boolean Conditional Logic

Expression	Interpretation
IF A AND B OR C	A and B must be true, or C must be true.

IF B AND A OR C	B and A must be true, or C must be true.

The interchanging of A and B with the standard logic gives different results whereas with the Boolean logic, it is the same.

All logic for a given RIT will be processed, using the same logic. The standard logic will be the non-Boolean type; however, the user may optionally specify the use of Boolean logic on the CREATE card (see 3.1.1).

3.2.9 Header Lines (HEADERn) and Trailer Lines (TRAILERn)

Header lines are specified following the FORMAT card. These are nondata lines printed at the top of each page. Header lines will normally contain system labels and other descriptive information concerning the report.

The general form for the header specification is:

	<u>conditional statement</u>
HEADERn	47 'ACTUAL LABEL'
	47 System label
	47 literal

n is the header level number

The labels or literals are located on the header line by specifying a low-order (rightmost) print position for each.

```
HEADER1 71 'UNCLASSIFIED'

HEADER1 103 'PAGE'

HEADER1 110 PAGENO
```

These specifications will cause a line to be formatted with the terms UNCLASSIFIED in position 71 and PAGE in position 103. The third example for HEADER1 demonstrates

the use of system label (PAGENO) on header lines. The contents of the system label PAGENO will be printed at the specified position.

When using system labels, sufficient space should be allowed on the line for the entire system field since no overlap is permitted between specifications. For example: the print position specified for the system label CLASSIF can never be less than 32 since the length of this field is 32 characters. If header lines are conditioned, the parameters of the conditional statement cannot be file fields. The conditional statement must be the first specification for that line if the output of the entire line is to be conditioned.

```
HEADER1 IF PAGENO LT 10

HEADER1 98 'INSERT FOLLOWING UNIT CHANGES'
```

This example would condition the first header line on the basis of the contents of the system label PAGENO. If the page number were less than 10, the phrase INSERT FOLLOWING UNIT CHANGES would be printed. The assumption was made for this example that the PAGENO card had a value of nine and therefore, the phrase would be printed once at the beginning of the report.

The previous example illustrates the use of the conditional statement at line level by which the output of the entire line is conditioned. It is also possible to have portions of lines conditioned as well. If two reports were to be produced from one RIT and the header line for each report differed on the basis of the date entered at execution time, they might be specified as follows:

```
HEADER1 85 'FISCAL YEAR 1955' IF OPDATE 8/11 EQ '1955'

HEADER1 85 'FISCAL YEAR 1964' IF OPDATE 8/11 EQ '1964'
```

The specification of trailer lines follows the same rules that apply to header line specifications. The card type is TRAILER. The positioning of the trailer lines is such that there is a blank line used as a separator between the last body line and the first trailer line or as otherwise specified by the user in the RIT. The user may

control the placement of the trailer lines by using the SPACE and SPACE...BEFORE statements. Thus if the user wants the first trailer line to immediately follow the last data line on a page, SPACE 1 BEFORE TRAILER1 should be specified.

3.2.10 Label Lines (LABELn)

It is frequently necessary or desirable to place similar nondata lines in the body of the report containing column identification or some other supplementary information. The general form for these label line specifications is essentially the same as those for header lines.

	Conditional Statement
	47 'ACTUAL LABEL'
LABELn	64 system label
	75 literal
	80 SORT 4/7

n is the label level number.

All of the discussion of header lines is applicable to label lines with these additional specifications:

The parameters of the conditional statements may be file fields.

```
LABEL1 IF CNAM CHANGES

LABEL1 14 'COUNTRY'
```

This example would cause the label line to be put out only if the data field called CNAM changed from a previous value.

These conditional statements can also be used in connection with individual phrases or portions of the line as explained with header specifications. Care should be taken when using the conditional statements on portions of lines because if all portions of a line were conditioned as

in the following example and none of the conditions were satisfied, a blank line would result.

```
LABEL1 47 'COUNTRY' IF CNAM CHANGES

LABEL1 59 'NATO' IF CNAM EQUALS 'FRANCE'

LABEL1 90 'POSSESSED' IF POP GE 0
```

For label lines associated with lines containing periodic data, two options are available. If the label line were to be repeated for each physical line containing the periodic subset(s) until the end of the set(s) is reached, the specification would be any of the following:

```
LABEL1 PERIODIC

LABEL1 SET(b)

LABEL1 PER
```

This causes the label to print whenever sets are active (PER) or when the particular set (SET(b)) is active.

If the line is to appear only once prior to the periodic line, the entry PERIODIC or PER would be omitted.

Both line conditional statements or the periodic statements must be specified before the specifications for the contents of the label line.

Label lines must be associated with data lines so that the label lines appear only when the line is printed. Each label must have a level number of one to three digits, and labels and data lines must follow one numbering sequence. A label preceding a data line and having the same number as the line will be printed only when the line is printed.

In the example:

```
LABEL1 ..........

LINE1 ..........

LINE2  ..........
```

```
LABEL3 .........

LINE3 ..........
```

LABEL1 and LABEL3 would be printed only if LINE1 and LINE3, respectively, were printed. It is not necessary that the numbering sequence be concurrent, only that it be in ascending sequence.

Two or more label lines may be associated with one data line by adding an alphabetic character to the level number.

```
LABEL2A 17 'DATE'

LABEL2B 19 'DAY MO YR'

LINE2 19 DATER
```

The previous example illustrates two label lines associated with one data line and would be printed as:

```
DATE

DAY MO YR

30 10 64
```

This technique may be used only with body label lines (those specified by LABEL) and, since they pertain to one data line, should be treated logically together. Thus, each set of label lines (those with the alphabetic level indication) should have only one set of conditional statements and only one periodic mode statement even though more than one physical line is involved. To indicate that the conditional statements or periodic statements pertain to all the labels within the group, the specification should be given as:

```
LABEL2A IF ....

LABEL2A .....
```

LABEL2B.....

LINE2 .....

The LABEL2A condition pertains to all the labels tied to the next line. Label lines have an inherent condition in that they appear only when their associated data lines appear. Thus, any conditional statement placed on label lines is subordinate to the output of the data line. OP checks the body lines counter automatically to see that all the labels and at least one occurrence of the line with the same line number can be printed on a page, otherwise OP ejects to a new page.

3.2.11 Data Lines (LINEn)

The primary emphasis in the specification of a RIT is the display of file fields with any editing, conversion, calculations, totals, or counts to be performed on the data.

Contents of a Data Line

The contents of a line may be either fields from the data record, results of calculations, literals defined at compilation time, system labels, self defining literals or work areas passed from the retriever.

Literals are defined as being either alphameric (bound by quote signs and up to 52 characters in length), binary, or numeric. A numeric literal may be either signed (+ 10) or unsigned (10). Work areas passed to OP by the retriever are defined by set association, user name, output length, and modes.

For printed reports, these specifications are given on line cards. The general format for these expressions is similar to the cards previously discussed. These specified lines are repeated for each record of input unless conditionally excluded.

Line Level Conditional Statement

```
            47 file field
LINEn       65 literal
            78 'ACTUAL LABEL'
            action statement
```

n is the line level number.

A line level number may be any 1- to 3-digit number which is not a duplicate of any other line level number. Duplicate line level numbers will cause assembler errors.

All output is specified by indicating the low-order print position on the line, followed by the field mnemonic, literal name, system label etc.

```
LINE1 24 UNIT

LINE1 38 LOC
```

It should be noted that a line specification does not necessarily mean one physical line of printing per record. If periodic data is specified, many lines may be printed from one line specification.

Variable data fields are indicated by specifying the high- and low-order print position on the line followed by the field mnemonic.

```
LINE1 21-120 REFER
        or
LINE1 21/120 REFER
```

One hundred characters from the variable field REFER will be printed per line until the variable field is exhausted.

Conditional Statements on a Data Line

Data line specifications may be conditioned as a whole or may be conditioned by portions.

```
LINE1 IF UNIT CHANGES

LINE1 24 UNIT
```

The above example demonstrates the conditioning of an entire line. Any specifications for LINE1 that follow would print only if the file field UNIT changes.

```
LINE1 24 CNAM IF CNAM CHANGES
```

This example shows the conditioning of only one portion of a line (field level conditional). The data contents of the field CNAM will print only when the field changes. Other specifications for this line will not be affected by this condition.

```
LINE1   IF CNAM CHANGES
LINE1A  40 CNAM
LINE1B  40 MEQPT
```

In the example, if CNAM does not change neither LINE1A nor LINE1B will be processed. However if it does, both will be processed. Note that if labels are associated with the LINE1 or the LINE1A or LINE1B they are dependent also on the LINE1 condition.

```
LINE1 IF ...

LABEL1A ...
LINE1A ...

LABEL1B ...
LINE1B ...
```

Labels and lines will be processed or not processed depending on the LINE1 condition.

3.3 Specifications for Printed Output

The following sections describe extended use of the processing and format specifications that do not appear in the simple output format. Specifications for punched card and magnetic tape reports are described in sections 3.4 and 3.5.

3.3.1 Title Lines (TITLELINEn)

The specification of a TITLELINE enables the user to give a title to the report. Title lines will be printed once per report and will appear at the beginning immediately following the header lines and may contain the same type of information as the header lines.

```
TITLELINE1 40 'THIS REPORT WAS'

TITLELINE1 53 'COMPLETED ON'

TITLELINE2 50 OPDATE
```

3.3.2 Format Control (SPACE, EJECT, SKIP)

Spacing

The normal spacing for a report can be indicated on the FORMAT card as previously discussed. This spacing should be the predominant spacing throughout the report. Special spacing to replace the normal spacing or for special conditions may be indicated by the SPACE card. The general format of the SPACE card is:

```
                conditional statement
SPACE n
                format control statement
```

Normally, the conditional statement will not be necessary and the spacing action will be specified as:

```
SPACE 5
```

This spacing will override the normal spacing (specified on FORMAT card) whenever the line (label, header, title lines, etc.) preceding this SPACE card is printed.

The conditional statement can control the spacing action as in the following example:

```
LINE1 ......

SPACE 2

SPACE 5 IF MEQPT CHANGES

LINE2 ......
```

This illustration provides for double spacing unless the field MEQPT changes; then, the spacing would be four spaces.

Format control statements have some limitations. If spacing between two periodic lines is desired to be different from spacing of different subsets of the same line, a label line must be inserted for the second periodic line and spacing must be conditioned on the printing of the label line.

The following specifications:

```
LINE1

SPACE 2

SPACE 3 BEFORE LINE2

LINE2
```

would cause triple spacing between the printing of each subset of line 2 and not just between the last subset with line 1 specifications and those with line 2.

This may be properly handled by:

```
LINE1

SPACE 2
```

```
SPACE 3 BEFORE LABEL2

LABEL2

LINE2

SPACE 2
```

If spacing is desired between records, another format control statement is used:

```
SPACE 7 BETWEEN RECORDS
```

This type of spacing expression will cause the specified spacing to take place only between each data record. The specifications must immediately follow the last data line for the record. Each of the above examples could be conditioned; however, if the action is conditioned, the conditional expression must be placed last.

Whenever a format control statement includes the conditional IF.....COMPLETE, the statement must include the BETWEEN RECORDS modifier. For example:

```
SPACE 3 BETWEEN RECORDS IF CNAM COMPLETE
```

EJECT Statement

Another method of controlling the format of a report is the EJECT statement. This statement will cause the program to eject to a new page when the expression is encountered, printing any trailer and header lines before proceeding to the next specification.

```
                conditional statement
EJECT
                format control statement
```

The above example illustrates the general format of the EJECT statement. If unconditional, the action will be taken each time the statement is encountered. The conditional statement can be used as in the following example.

```
EJECT IF BODYLINES LT 5
```

This statement will cause the page to be ejected if the number of lines remaining on a page is less than five.

The two forms of the format control statement that can be used with EJECT statements are:

EJECT BEFORE LABEL5

EJECT BETWEEN RECORDS

If the latter is used, it must immediately follow the last data line for the record.

Format Control Using the Carriage Control Tape (SKIP)

In addition to these methods of controlling the format of a report, the carriage control tape on the printer may be used. The printer may be skipped to any channel on the carriage control tape with the use of the SKIP specification card. The card has the format:

```
                format control statement
SKIP TO n
                conditional statement
```

The n is the number of the channel to which the printer will skip. This action takes place immediately before the next line card is printed. Care should be taken when using this option, since the body lines counter is not decremented as the result of the skip action. The body lines counter must be adjusted through the RESET card if the format is controlled by both skip and spacing action. Also, the carriage tape must be properly punched.

The SKIP specification may have conditional statements and format control statements as previously discussed with the SPACE and EJECT specifications.

3.3.3 Overflow Line Specifications (OVERFLOWn)

The OVERFLOW operator is a feature of the Output Processor that executes when bodylines have been exhausted for the previous page of output. The OVERFLOW block serves a similar function as a LINE block, however, it may execute

only at the beginning of a bodyline section after an overflow condition has occurred. The format of the OVERFLOW statement is:

```
                                  Conditional Statement
                                  47 file field
OVERFLOWn                         47 literal
                                  47 System label
                                  47 'actual label'
                                  47 sortkey
                                  Action statement
```

An OVERFLOW level number may be any 1- to 3-digit number. A line level alphabetic is an option and may be used to control multiple lines of output within a level number.

OVERFLOW blocks are automatically conditioned to execute on a page which has been reached by overflow from the previous page. Output may be further conditioned to respond to the full range of tests allowed for LINE statements.

References to data file fields/groups, previously defined literals, system labels, self-defining literals, sortkeys and action statements are allowed on OVERFLOW lines.

OVERFLOW blocks will not function if one of the following conditions occur:

a. The first page of output

b. The first line block or label line is in the process of executing or to be executed when an overflow condition has been reached

c. The current page reached by the process of the EJECT format control operator.

OVERFLOW statements appear in a RIT before the first LINE block reference and after the last HEADER statement or TITLELINE statement. Format control operators (SPACE and EJECT) may precede or follow OVERFLOW statements, however, caution should be taken when the EJECT operator is used.

The following statements (in part) are an example of an OVERFLOW application:

```
HEADER1 74 'FINANCIAL REPORT'
SPACE 5
TITLELINE1 80 'BUDGET ACCOUNTING BY SERVICE'
SPACE 5
OVERFLOW1 8 'SERVICE='
OVERFLOW1 14 SERV
OVERFLOW1 26 '(CONTINUED)'
SPACE 2
LINE1 IF SERV CHANGES
LINE1 8 'SERVICE='
LINE1 14 SERV
SPACE 2
   .
   .
```

In the previous example OVERFLOW1 will be ignored for the first page of output and LINE1 will execute when the first record is processed and when the SERV field value changes in records that follow. When an overflow condition is sensed OVERFLOW1 will execute at the beginning of the bodyline section of the new page if LINE1 is not scheduled to execute. After OVERFLOW1 completes it execution the RIT will continue to function at the point in the program where it was interrupted when the overflow condition was sensed.

3.3.4 Definition of an Area (DEFINE)

The DEFINE card can be used to specify the length of an area and give it a name. These areas are useful for: (1) work areas for computations, (2) result areas for computations for which the system-produced lengths are inadequate, (3) a literal with a specific value or name that might be used repeatedly, and (4) work areas passed to OP by the retriever.

Additionally, the DEFINE card may be used to give a name to a repeatedly used edit word, to an array literal and to a matrix literal.

```
DEFINE WORKA 4

DEFINE BUCKET B 8

DEFINE TITLE 'THE OUTPUT SYSTEM USERS MANUAL'
```

The first example will cause a work area of four digits to be created and associated with the term WORKA. It will be treated as a numeric field for all output system functions.

The second example will cause a work area of one full binary word to be created. This work area would be used for calculation of binary data fields. The 8 specifies the maximum output length.

The third example will define the literal THE OUTPUT SYSTEM USERS MANUAL and the phrase can be referenced with the term TITLE. Literals defined in this manner will be considered alphameric. Embedded quotes or ampersands are illegal.

The names assigned to these literals can be any name except those with special meaning to the output program.

The literals defined in the above manner will be treated as fixed fields. They can be associated with particular periodic sets or as periodic literals in general as in the following examples:

```
DEFINE SET 1 AREAONE 4

DEFINE PERIODIC TDY 'TDY DATE'

DEFINE PER STATUS 'TDY'
```

The first example defined a 4-digit work area, AREAONE, for periodic set one. The next two examples are two alternative means of defining TDY and STATUS as periodic literals. A periodic literal for a specific set will be repeated (if specified to be printed on a line) until the fields in the set have been exhausted. A general periodic literal will be repeated until all periodic sets specified on that line are exhausted.

Note: The maximum length of an alphameric literal is 52 characters. The maximum length of a binary literal is 10 characters. The maximum length of a numeric literal is 999 characters.

The maximum output length of a numeric literal output by OP is 15 bytes. This same length restriction applies to the numeric literals used in arithmetic operations. Partial field notation may be used to address and output portions of numeric literals longer than 15 characters.

Work areas containing data values generated by the Retriever FUNCTION Operator subroutines are available to the Output Processor for display or further processing. These work areas are referenced by a user-assigned name. There are five fullword (4-byte) work areas associated with each set. The system labels for these work areas are WORK1 through WORK5. Specification of these areas is made as follows:

```
DEFINE   SET n     name    WORKm    c    t
```

where

DEFINE - Identifies the statement

SET n - Identifies the set number associated with the work area

name - User-assigned name for reference to the work area

WORKm - Identifies the individual work area (WORK1 through WORK5)

c - Number of characters to be output

t - Type code which identifies the mode of data in the work area (A-alpha, B-binary, C-internal form coordinate, D-decimal; if omitted, the mode is assumed to be binary).

Alpha and decimal work areas can be one to four bytes in length. If less than four is required, bytes will be used starting at the left (high-order) and dropped from the rightmost (low-order) side. Coordinate fields can be five six, seven or eight bytes in length, corresponding to the standard conversion length of latitudes and logitudes from internal to external format.

Reference to the work area is accomplished by using the name assigned in the DEFINE statement as shown in the following example:

```
DEFINE   SET Ø  DIST   WORK1   5   B
LINE1    50   DIST
```

Assume the retrieval component has caused a value to be generated in WORK1 for set Ø (fixed set). The DEFINE statement assigns the name DIST to the work area and specifies the area contains binary data. The output length is specified as five characters. The LINE1 statement specifies that the work area, referenced by the name DIST, is to be printed in position 50.

The user should note that work areas should be thought of as file data fields in terms of the manner in which they

are processed. When they are associated with periodic sets, values are passed by retrieval only for flagged (qualifying) subsets. The work area value of binary zero will be used by OP for nonflagged subsets when the RIT specifies processing of periodic data in the ALL mode (see section 3.3.13). Specification of these work areas should not be made in RITs which are to be run directly against a data file, since no work areas will exist.

Defining named edit masks within a RIT provides the capability to repeatedly use the edit literal on output specifications of numeric fields. The specification of a named edit mask is as follows:

```
DEFINE EDIT XXXX 'editmask'
```

where

DEFINE - identifies the statement type

EDIT - is the keyword parameter designating the literal as an edit mask.

XXXXXXX - the user designated name for the edit mask (up to seven characters).

'editmask' - is the user designated edit mask

Reference to the edit mask is accomplished by using the name assigned in the DEFINE statement when printing and editing a numeric field as in the following example:

```
DEFINE EDIT LEDIT 'bbb0bb'

LINE1 30 PERS LEDIT
```

The number of replaceable characters within defined edit masks must be equal to the length of the field to be edited.

Addressing portions of fields and literals is discussed later in section 3.3.8. An alternative to partial field notation, which may be used when referencing numeric literals, is referred to as array definition and processing. Basically, an array is defined in the same manner as a numeric literal. The difference is that there is a

reference to the number of occurrences or repetitions of the literal. An array is considered as one row of n columns or occurrences. Thus, the user may directly or indirectly, with the use of counters, reference any segment of the array by its occurrence number. The definition of the array literals may take the following formats:

```
DEFINE LIT1 (X) n
DEFINE LIT2 (X) B n
```

In the first example, the array LIT1 is defined to have X occurrences with the length of each occurrence being n. Each of those occurrences is treated as a numeric literal for all output functions. Note the parentheses around the occurrence number; they must be present.

The second example, once again, has X designating the number of occurrences of length n for the array literal LIT2. B defines the literal as being binary and maintained as binary fullwords.

Matrix literals are used in conjunction with the elementary matrix capability in OP. The matrix definition can be thought of as an advanced array in that a matrix has all of the characteristics of an array. As previously stated, an array consists of one row with n columns or occurrences. A matrix is composed of one or more rows with n supportive occurrences or columns. With arrays it was necessary to specify only the number of columns. With matrices the numbers of both rows and columns are required. The definition of a matrix is as follows:

```
DEFINE MATA (X,Y) n
DEFINE MATB (X,Y) B n
```

In these examples X represents the number of rows in the matrix while Y represents the number of columns or occurrences. The output length of each occurrence is represented by n. In the second example B designates the matrix occurrences to be stored internally as binary fullwords.

3.3.5 Storing Data for Later Use (MOVE)

The MOVE expression is used to transfer data either from a data record to a defined area of the RIT, or from one defined area of the RIT to another defined area of the RIT. Data being moved replaces whatever data was formerly contained in the receiving literal. The contents of the field or literal being moved remain unchanged.

MOVE may be specified as a level one or as a level two expression and has the following general format:

```
MOVE parameter TO literal
```

The parameter may be a file field/group, a previously defined literal, a self-defined literal, an array, or a matrix. The literal may be a previously defined literal, array, or matrix.

Subroutine or table conversion is possible for a field or literal which is not binary. The following format is used for subroutine/table conversion:

```
MOVE parameter SUBRT submove TO literal
MOVE parameter TABLE tabname TO literal
```

The following conventions or restrictions apply to the use of the MOVE expression:

a. The output length of the literal must be the same as the output length of the parameter unless partial field notation is specified, subroutine conversion is specified, the parameter is an entire array or matrix, or the parameter is a coordinate mode field.

b. Partial field notation is accepted for the parameter and/or the literal whenever it is valid (see section 3.3.8, Addressing Portions of Fields). When used, the lengths of the specified portions of the parameter and/or the literal must be equal.

c. Subscript notation is accepted for the elements of a matrix or an array (see section 3.3.8, Addressing Portions of Fields).

d. When an entire matrix is specified as the parameter (i.e., no subscripts are provided after the matrix name), the literal then must also be an entire matrix. The elements of both matrixes must have the same output length, and the total number of elements in both matrixes must be the same.

e. When an entire array is specified as the parameter (i.e., no subscript is provided after the array name), the literal then must also be an entire array. The elements of both arrays must have the same output length, and the number of occurrences in both arrays must be the same.

f. When subroutine or table conversion is specified, the length of the literal, or the length of that part addressed by partial field notation, must be equal to the length of the maximum function.

g. A variable field or set may not be specified as the parameter.

h. If the parameter is numeric mode, the literal must also be numeric mode.

3.3.6 Conditional Exclusion of Data Records (OMIT)

Data records may be conditionally excluded from a report or from one of the output media with the use of the OMIT specification. This statement enables the user to be selective in the processing of records for any one of the specified output media, or both.

The production of an index, on punched cards, is an example of its application. The following specifications illustrate the manner in which a report would be specified where only the data on units in the country of Germany is to be included in the printed report, and an index of the pages where each location began is to be punched.

```
FILE TEST360

OMIT IF CNAM NOT EQUAL 'GERMANY'

FORMAT PRINT
```

```
-------
-------

FORMAT PUNCH

OMIT IF LOC NOT CHANGE

CARD1 30 LOC

CARD1 78 PAGENO

-------

END
```

The OMIT statements in the previous example eliminate the necessity for having each specification conditioned with the same statement.

The correct sequencing of these statements is important when they are used to select data records for processing. The example illustrates the case where only one file is contained in the report. When used in this instance, the statements placed between the file card and the format card pertain to all formats; the statements placed after a format card pertain only to that format.

If conditioning on more than one field is needed to limit the number or type of records to be processed, then multiple OMIT statements must be used to assure expected results.

```
OMIT IF SERV NE 'J'
OMIT IF UUIN NE '00001' AND UUIN NE '00002'
```

The above would result in only the records with SERV equal to 'J' and UUIN equal to '00001' or '00002' being processed. All others would be omitted.

Care must be taken in the meaning of the TOTAL OF_____and COUNT OF_____statements when used in conjunction with conditional exclusion of data records from

a report (the OMIT statement). This is discussed in section 3.3.17 "Expanded Arithmetic Operations."

3.3.7 Stopping the Report Conditionally (STOP)

The production of the report can be stopped at any time by the STOP statement. This conditional statement should be the first statement following the FILE card for the file to which it pertains.

```
FILE TEST360

STOP IF CNAM EQUALS 'ITALY'

-------

-------

-------
```

When the condition is satisfied, the report is terminated. In the previous example the report would be stopped when the country was ITALY. The record which contained ITALY would not be processed.

Care should be taken that this specification is placed in proper sequence.

```
FILE TEST360

FORMAT PRINT

FORMAT PUNCH

STOP IF CNAM EQUALS 'ITALY'

END
```

If this statement were placed as shown above, the record containing ITALY would be printed and, since the condition is satisfied, the report would then stop.

3.3.8 Addressing Portions of Fields (Partial-field Notation)

If a portion of a field or literal is desired for output or for use in other specifications, the portion would be indicated as follows:

```
FIELDA X/Y
```

where x is the high-order and y is the low-order position of the field.

This notation can be used in all specifications where previously defined literals and fields are referenced with the exception that a partial field cannot be specified when a system subroutine (one specified in the FFT) is to act on that field. Partial field notation is not allowed on variable fields and sets, or binary literals and fields.

```
LINE1 47 FIELDA 1/4

LINE1 38 TOTAL OF FIELDB 5/6

LINE1 80 FLDC 1-4 DIV FLDD 5/9
```

The above examples illustrate the specification. The partial fields are treated in every way as the whole field would be except as mentioned above. This notation has a special significance when printing a RASP answer set, and is discussed in the following section.

Another means of addressing portions of literals is associated with the use of array and matrix literals (3.3.4). With these types of literals, subscripts are used to point to particular segments or sections of the literal. These subscripts follow the user supplied literal name and are enclosed within parentheses. Subscripts may be any of the following: a self-defining numeric literal, a previously defined numeric literal or a numeric file field. In the latter two cases, it is the content of the field that is actually used for the subscripting.

An array, which can be thought of as one row of n columns or occurrences requires a single subscript

designating the proper occurrence of the literal. Matrixes are composed of one or more rows with n columns or occurrences and require a pair of subscripts that designate the row and column.

When portions of an array or a matrix are desired for output or for use in other specifications, these portions would be indicated in the following manner:

```
ARRAY2 (5)
ARRAY3 (CRT)
MATRIX4 (2,5)
MATRIX5 (3,SUBS3)
```

3.3.9 Addressing the RASP/QUIP Sort Key

The RASP sort key is variable length (a function of the accumulated length of fields designed by the user for sort requirements in the RASP query). The first two positions of the sort key contain the RASP query number (QUERYNO). The third character position in the sort key is not used. Sort key position 4 contains the first character of the user desingated sort.

Two procedures have been provided to access information in the sort key of the RASP answer records.

a. Partial File Notation

   LINE1 IF SORT 5/10 CHANGES

   The keyword for the sort key is SORT, and with the use of the partial field notation, any position of the user-designated sort can be referenced. Note, however, i.e. the user's responsibility to coordinate the specified sort in RASP and OP.

b. Named Fields of the Sort key

   The file name in the answer record can be referenced by the term FILENAME; the query number by the term QUERYNO. Either of these

two names can be used in the same way as file fields.

The QUIP sort key is similar to the RASP sort key because both sort keys are variable length and are a function of the accumulated length of fields designated by the user for sort requirements. However, the QUIP sort key contains no query number or leading blank positions, therefore, sort key position 1 contains the first character of the user designated sort.

The following example demonstrates the technique for partial field notation of a QUIP generated sort key.

```
LINE1 IF SORT 1/6 CHANGES
```

3.3.10 Edit Feature, Subroutines, and Tables (EDIT, SUBRT, TABLE)

Numeric file fields can be edited using the editing feature or converted by a conversion table or subroutine. In most cases these edit words, tables, or subroutines have been specified by the file designer in the File Format Table (FFT). A field name (print format only) with no subroutine/table/edit designation will default to the output subroutine/table/edit functions specified in the FFT. A fieldname followed by the word SUBRT, TABLE, or EDIT will do the same. The FFT designation can be overridden by the fieldname followed by ## (subroutine or table) or ' ' (quote, quote for edit). The output will be in file form or it can be overridden by #SUB/TAB#, SUBRT SUBNAME, TABLE TABNAME, EDIT 'editword', or the name of a defined edit word (3.3.4) and the data will be processed using the designated subroutine, table, or edit word.

Note: Subroutines, tables, and edits are not automatic on card and record specifications.

OUTPUT PROCESSOR (OP)

Example:

```
LINE1 20 SERV       )  Uses subroutine from FFT
                    )
LINE1 20 SERV SUBRT)

LINE1 20 SERV ##      Output will be in file form

LINE1 20 SERV #SUBX#  )
                      )  Output will be processed by
LINE1 20 SERV SUBRT SUBX) subroutine SUBX

LINE2 30 PLDTG      )  Uses edit pattern from FFT
                    )
LINE2 30 PLDTG EDIT)

LINE2 30 PLDTB ' '    Overrides FFT edit and output
                      data in file form

LINE2 30 PLDTB NAMEDIT      Override FFT edit and
                            uses defined edit word
                            NAMEDIT

LINE2 30 PLDTB '##-##-##'   Overrides FFT edit and
                            specifies edit pattern
                            to be used
```

When directly specifying the edit pattern to be used on a field, as in the last two examples above, care must be taken to ensure that any nonreplaceable characters to be inserted are accounted for when print positions are designated.

The following instructions will move a portion of a table function to print:

```
DEFINE LIT1 '########' (size literal to maximum length
of function.)

LINE3 MOVE RADTG SUBRT DTGOS TO LIT1
LINE3 16 LIT1 3/5
```

If RADTG equals 680307, 07MAR68 would be moved to LIT1 and LINE3 position 16 would contain MAR, thus rearranging a date field to make it more readable at output time.

Note: If there is an unsuccessful return from a user table or subroutine, the argument will be output by OP.

The following instructions will move a portion of a data field to a work area for subsequent editing.

```
DEFINE WORKA 6
LINE1 MOVE TRDTG 1/2 TO WORKA 5-6
LINE1 MOVE TRDTG 3/6 TO WORKA 1-4
LINE1 15 WORKA EDIT 'ØØ-ØØ-ØØ'
```

IF TRDTG equals 651215,
this procedure would print 12-15-65.

The logic might be extended in this way:

```
LINE1 9 'DEC' IF WORKA 1/2 EQ '12'
LINE1 15 WORKA 3/6 EDIT 'ZZ,ØØ'
```

and the printout would be DEC 15,65.

Note: Editing may be done only or numeric file fields or literals.

Note: Binary values cannot be passed through a TABLE. The results will be either the printing of blanks or meaningless characters. The following logic will accomplish the task successfully:

```
DEFINE NUMFLD 2
LINE1 MOVE BINFLD TO NUMFLD
LINE1 10 NUMFLD TABLE BINARY
```

3.3.11 Literals on Data Lines

Literals can also be specified on a line by the use of a bounded label (by quote signs) or by a label that has previously defined a literal.

```
LINE1 47 'ACTUAL LABEL'

      or

LINE1 50 LIT1
```

In the latter example, the term LIT1 has been previously defined with a DEFINE card.

When a literal and periodic data are to be printed on the line, the literal will be treated as fixed in that it will appear only on the first physical line. If, however, the literal is desired on every physical line containing periodic data until the periodic set(s) is exhausted, then the specification would be:

```
LINE1 47 'DAY MO YR' PERIODIC

          or

LINE1 47 'DAY MO YR' PER
```

It is also possible to tie the literal to any particular set so that the literal will be repeated only until that set is exhausted by specifying as in the following example:

```
LINE1 47 'DAY MO YR'     SET n
```

n is the periodic set ID.

Referencing array literals for output purposes is done by using the literal name followed by a subscript enclosed within parentheses. These subscripts may be a specific numeric literal value, the contents of a previously defined numeric literal or the contents of a named numeric file field. This can be shown by the following:

```
LINE10 30 LIT1 (2)
LINE10 59 LIT2 (SCRIPT)
LINE20 40 LIT2 (PERS)
```

where SCRIPT is a previously defined numeric literal and PERS is a numeric field from a data file.

When referencing the matrix literal for output purposes, the user must specify what portions of the matrix he wishes to output and where he wants the output in his formatted output line. To do this, the matrix literal name must be specified at an output position along with a set of subscripts, within parentheses, representing the desired row

and column. These subscripts, as with arrays, may be numeric literals, the names of previously defined numeric literals, or named numeric file fields. The following are examples:

```
LINE10 20 MATA (1,5)
LINE10 40 MATA (2,SUB2)
LINE10 60 MATB (SUB3,8)
LINE10 80 MATB (9,PERS)
```

It should be noted that when using previously defined literals or file fields it is the contents of these fields that determines the appropriate row or column.

In order to reduce the number of specifications required to output a matrix, a second method is available. It allows specification of each low-order output location chosen for the matrix literal on the output statement. In previous discussions of subscripting, subscripts represented one row and one column, thereby directing OP to a particular segment of a matrix. However, utilizing subscripts in association with multiple print positions, a start column and a stop column are specified in conjunction with a row or else a start row in conjunction with a column. The following are examples:

```
LINE10 20 40 60 MATA (X-Y,Z)
LINE20 20 40 60 MATB (A,B-C)
```

The first example shows that, starting with row X and stopping with row Y, the values found in column Z are to be printed in positions 20, 40, and 60. Giving the symbols X, Y and Z values of 1, 3, and 6 respectively, the following will be printed: in position 20, row 1, column 6; in position 40, row 2, column 6; and in position 60, row 3, column 6.

In the second example of multiple print position with matrices, the values in row A from column B through column C will be printed in the specified positions.

When using the start and stop subscript, the number of rows of columns may not exceed the number of print positions without generating an error diagnostic. This method of

matrix output specification reduces the number of OP statements required to output the matrix.

3.3.12 Periodic Information (SCAN, ALL, FIRST, LAST, SET)

No mode specification is necessary if the periodic mode (SCAN) for the report is to remain the same throughout. (This is essentially for a report received from RASP.) If, however, the mode desired differs or is not consistent within a RIT, then the mode can be specified on each line. Specification for each line containing periodic information to be printed in the SCAN mode may be as follows:

```
LINE1 SCAN
LINE2 SCAN SET n where n is the set ID.
```

The first of the two examples states that all periodic sets are to be printed in the SCAN mode for every reiteration of LINE1 (flagged subsets only). The second example can be used to tie the mode to a particular periodic set. If this were done, the specified set would be printed with that mode while the rest of the periodic sets would be printed with the normal file mode.

If a mode is not specified at structure time, SCAN is assumed.

To specify that the LAST n or the FIRST n subsets are to be printed the statements are:

```
LINE1 FIRST 5

LINE2 LAST 10

LINE3 FIRST 5 SET 5

LINE3 LAST 10 SET 6
```

The first example causes only the first five subsets of each periodic set to be processed while printing line 1. The second example causes the last 10 subsets of each set to be processed for line 2. The last two examples tie the

specifications to specific periodic sets. These specifications override any assigned mode.

If the mode for retrieval specified for a periodic set is other than SCAN and all subsets are wanted in the report, the specifications would be:

```
LINE1 ALL
```

or

```
LINE1 ALL SET n
```

The first example will cause all subsets to be printed for all periodic sets specified on the line. The second example ties the specification to a periodic set. This specification is not necessary when printing a report directly from the file, in which case all subsets are printed.

The periodic mode specified in this manner will affect the output in this line only. Thus, if fields from the same periodic set were placed on two lines, each line would have to have the mode specified. There is no need to keep the mode for a set consistent throughout the report.

With periodic information it is often required or preferred to have the same field (or fields) from more than one subset printed on a line. This is done by specifying each low-order output location chosen for that field on the line.

```
LINE1 10 20 30 FIELD
```

The above example would place the specified field from three subsets on line 1 in the positions indicated. The line would be repeated in this mode until the set was exhausted. If more than one field from a subset is specified, it is important that all fields from the same set be specified in the same way; that is, if more than one subset per line is indicated for a field, the specification for another field in the same set must be the same.

There are two main ways in which fields from the same periodic set can be displayed on more than one line. The first would be on the basis that each line is repeated until the set is exhausted and then the next line is repeated until the set is exhausted. If a periodic set contains fields, A, B, C, and D, the specifications might appear as:

```
LINE1 10 FLDA

LINE1 20 FLDB

LINE1 30 FLDC

LINE2 10 FLDD
```

The format would appear as:

```
Subset 1   A    B    C

       2   A    B    C

       3   A    B    C

       1   D

       2   D

       3   D
```

It might be desirable to display the above set in the following manner:

```
           A    B    C

           D

           A    B    C

           D

           A    B    C

           D
```

Since the lines can be considered as one group, this pattern

OUTPUT PROCESSOR (OP)

would be specified as follows:

```
LINE1A 10 FLDA

LINE1A 20 FLDB

LINE1A 30 FLDC

LINE1B 10 FLDD
```

The only mode limitation that must be placed on this method of specifying periodic sets is that the mode for each set and the number of subsets per line must be the same in each line.

If a periodic mode is specified for a series of lines containing periodic information from the same set, it would be specified as follows:

```
LINE1 SCAN

LINE1A 10 FLDA

LINE1A 20 FLDB

LINE1A 30 FLDC

LINE1B 10 FLDD
```

Note: Refer to conditional statements in section 3.2.8 for further restrictions.

3.3.13 Associated Label Lines

Each line in a group of lines may have associated labels, or the entire group of lines may have one or more labels associated with it.

```
LABEL1A 47 CLASSIF

LABEL1B 50 'ACTUAL LABEL'

LINE1B 10 FLDA
```

```
    LINE1B 20 FDLB

    LINE1C 30 FLDC
```

In the above example, the label lines 1A and 1B are associated with the first data line (LINE1B) that follows. The level sequence given in this example follows the rule which states that when proceeding from a label line with an alphabetic level to a data line (LINE) with an alphabetic level, the level of the data line must be the same as or greater than the preceding label line.

```
    LABEL1A PERIODIC

    LABEL1A 26 CLASSIF

    LINE1A .......

      .

      .

    LABEL1B PERIODIC
    LINE1B .......
      .
```

This example illustrates a group of lines which has label lines associated with each line. Note that since the labels are to be repeated with the data lines each time they are printed, they have been specified as periodic.

Each of the label lines or groups of label lines may be conditioned provided that the conditional statement appears as the first specification for a series of contiguous label lines (the first alpha label line of a group). The output of these lines can be conditioned with the use of the conditional statements described previously.

The only restriction placed upon line level conditional statements (those which condition the output of the entire line) is with periodic lines. A periodic line containing more than one subset per line cannot be conditioned on a field from a periodic set. Also, as a general rule, lines with alphabetic level indication (LINE1B) should not be

conditioned. If alpha labels and lines are used, the only specifications which should be placed on nonalpha lines (LINE1) are line level conditions and action expressions. No print specifications should be used on nonalpha lines when the mixed mode (alpha and nonalpha statements in a block) is used.

3.3.14 Variable Information

Variable information such as remarks must be indicated in this format:

```
LINE1 8/100 REFER
```

This specification provides for data from a variable set or variable field to be printed from position 8 through 100. The information will be repeated in this format until exhausted.

No conditioning is permitted at level 4 of a line, card, or record statement specifying variable data. However, the statement may be conditioned at level 2, for example:

```
LINE1  IF VS27 GT 0
LINE1  8/100 REFER
```

Note: Partial-field notation may not be used on variable information.

Note: Variable sets may not be referenced in the BREAKLINE section.

3.3.15 Action Statements (TOTAL, COUNT, ADD, SUB, MUL, DIV, RESET)

Simple totals and counts of data fields can be specified on line cards as follows:

```
LINE1 50 TOTAL OF MEORC

LINE1 60 COUNT OF MEQPT
```

The two examples above would print the total number of ready equipment (right-justified) and the count of the equipment ID.

OUTPUT PROCESSOR (OP)

These statements specify the result of the action, not the performance of it. The total or count would be one which had accumulated since the beginning of the report or since its last printing. This would be a total or count by record unless conditional printing is specified. The actual totaling or counting cannot be conditioned, only the printing. Totals and counts are cleared when printed.

TOTAL OF accumulates a sum for the occurrences of a fixed numeric or of a periodic numeric field belonging to sets that have been flagged. COUNT OF tallies the occurrences of a fixed-length field, either fixed or periodic. Variable-length fields and variable-length sets (VSETS) may not be counted.

Simple calculations can be specified by:

LINE2 100 Factor, Operator, Factor, Operator,
Factor, etc. This might look like:

LINE2 100 Field PLUS Field MINUS Literal

The calculation would be performed left to right and the result will be printed in position 100. The factors may be field names, numeric literals, or a literal that has previously been defined. The operators that can be used are:

Addition	Subtraction	Multiplication	Division
ADD PLUS	SUB MINUS	MUL	DIV
+(12 punch) or (12-6-8 punch)	-(11 punch)	*(11-4-8 punch)	/ (0-1 punch)

The results of totals, counts, or calculations can be conditionally printed on a line by the use of conditional statements.

LINE1 50 COUNT OF MEQPT IF CNAM EQ 'GERMANY'

LINE1 60 MEPSD MINUS MEORC IF MEPSD GE MEORC

The above examples will cause the count of MEQPT to be printed for each record for the country of Germany and the number not ready (MEPSD MINUS MEORC) if the number possessed is greater than the number ready.

The results of totals, counts, or calculations of this type can also be edited or be acted upon by a subroutine or table.

```
LINE1 80 TOTAL OF SEAUTH EDIT 'Øbb,bbb'
```

This example would cause the result of the totaling of SEAUTH to be edited with the edit word provided.

Computations with Literals

Another method of specifying additions or subtractions to or from predefined literals is by the following:

```
LINE1 ADD parameter TO literal

LINE1 SUB parameter FROM literal
```

The parameter may be numeric fields or numeric literals. The literal must be either a system label or a numeric area which has been previously defined. These statements can be conditioned. Note, that like the MOVE, when the calculation is designated on a LINE, care must be exerted to avoid the printing of blank lines. The ADD and SUB statements may be specified as separate statements similar to the MOVE when it is not necessary or convenient to perform these functions on a line.

RESET

The RESET card can be used to reset any arithmetic result area or numeric literal to a particular value. It may be placed anywhere in the specifications. The general format of the card is:

```
RESET area conditional statement

        or

RESET area TO_______ conditional statement
```

The first statement is used when resetting a numeric literal (up to 999 characters in length) to zero. The second statement will reset the area to the value specified. The resultant area will have true sign and leading zeros. The reset statement may also be used on a LINE card or RECORD card. Additionally, the RESET statement may be used in conjunction with array and matrix literals. These literals are described in section 3.3.4. In these instances, the basic concepts remain the same with the use of subscripts providing the only difference.

The following RESET statements are appropriate for arrays:

```
RESET LIT1
RESET LIT1 TO 5
RESET LIT1 (2)
RESET LIT1 (SCRIPT) TO 4
```

The first statement above would reset all occurrences of the defined array literal, LIT1, to zero, and the second would reset all occurrences of the literal to 5. The third statement would reset the second occurrence of LIT1 to zero, and the fourth would reset the occurrence of LIT1 designated by the contents of the literal SCRIPT to 4.

The means of restoring matrix literals to an initial value is similar to that used with arrays except that matrixes require two subscripts to designate a row and a column. The following show how restoration or reinitilization is designated:

```
RESET MATA
RESET MATA TO 5
RESET MATA (1,5)
RESET MATA (1,3) TO 4
```

The first example resets the entire matrix (each member of each row and column) to zero while the second resets each member of the matrix to 5. The third and fourth examples show resetting with the aid of subscripts. The third example illustrates the member at row 1, column 5 being

reset to zero. The fourth example resets the member represented by row 1 and column 3 to 4.

The RESET card is useful when resetting numeric system labels.

```
RESET BODYLINES TO 10

RESET PAGENO
```

The first example will reset the body line counter to ten, while the second example will reset the page counter to zero. The ability to reset these system constants allows more flexibility of format control under the control of user data. For instance, the page number could be reset to a value, based upon a change in a control field, to produce separate volumes of a report.

Since the page counter is incremented after the trailer lines are printed, the location of the page number will determine when and to what value the system label PAGENO is to be reset.

Page Number on Header Line

The system constant PAGENO should be reset to one less than the desired value and, if an EJECT statement is present with the same condition, placed on the line preceding the EJECT card.

```
FORMAT .......

HEADER1, 47, PAGENO

LINE1 ........

LINE1 RESET PAGENO IF SERV CHANGES AND SERV EQ 'F'

EJECT IF SERV CHANGES AND SERV EQ 'F'

TRAILER1
```

Page Number on Trailer Line

PAGENO should be reset to the desired value and placed after at least one data line specification.

```
FORMAT .......

HEADER1 ......

LINE1 ........

RESET PAGENO TO 1 IF SERV CHANGES AND SERV EQ 'F'

LINE5 ........

EJECT BETWEEN RECORDS IF SERV COMPLETE

TRAILER1 47 PAGENO
```

Reset Resultant Area From Computations

Results of calculations, totals, and counts may be reset.

```
FORMAT .......

LINE1 COMPUTE TOTLMIL EQ TOTAL OF MEPSD IF MECLQ
      EQ 'C'

LINE2 IF SERV COMPLETE

LINE2 47 TOTLMIL

LINE2 RESET TOTLMIL
```

The example illustrates how the total of class C equipment possessed might be specified.

In these examples, the conditional statement IF_____ COMPLETE has been used. The use of this expression is discussed in this section under the heading "Summary Reports."

3.3.16 Result Areas (Standard Lengths)

Arithmetic operations may be performed on integer values that appear in the form of self-defining literals, defined work areas, data file fields or OP defined literals. Arithmetic operations include action expressions (TOTAL and COUNT), simple calculations (ADD 1 to WORKX) and expanded arithmetic operations (COMPUTE WORKZ EQ 1 + WORKZ). The result of each arithmetic operation is stored in a work space whose length has been calculated by OP based on the length of the factors involved in the calculation. The standard length of the result area is determined as follows:

TOTAL	Twice the length of the field being totaled
COUNT	Six digits
COMPUTE	Addition - length of longest factor plus one
	Subtraction - length of longest factor plus one
	Multiplication - sum of the lengths of the two factors
	Division - equal to the length of the dividend.

Examples of result lengths calculated by OP:

(a) LINE1 40 TOTAL OF PERS

In this example PERS is a 6-character numeric file field. The result of the totaling operation will be stored in a 12-character OP defined area and printed on a line with the right-most character in position 40.

(b) LINE1 40 COUNT OF SERV

In this example SERV is a 1-character alpha field. A count of all records that contain this field will be performed and the result of the count will be stored in a 6-character OP defined result area and

printed on a line with the right-most character in position 40.

(c) LINE1 50 WORKA + WORKB + WORKC

In this example WORKA has a defined length of 3 characters; WORKB has a defined length of 7; and WORKC has a defined length of 5. The result area calculated by OP will be the length of the longest factor plus one, i.e. 8 characters. The result will be printed on a line with the right-most character on position 50.

(Note: In the previous examples, the OP defined work areas will be initially set to zero prior to arithmetic processing. The work areas will be restored to zero when results are published.)

(d) LINE1 COMPUTE WORKX EQ PERS * 100 / WORKA

In this example the length of the result area, WORKX, will be calculated by OP based on the length of factors and use of arithmetic operators in the arithmetic expression. Calculation of factors in an arithmetic expression is performed by pairs of factors from left to right (no parenthetical expressions permitted). The file field PERS (length 6 characters) is multiplied by the self-defining literal 100 (length 3 characters). The combined length of the two factors creates an interim result area of 9 characters. WORKA is a previously defined work area with a length of 7 characters and is the divisor in divide expression in which the 9-character interim work area is the dividend. The quotient or result of the divide operation will be stored in a work space the length of the dividend (9 characters). The result area, WORKX, has not been explicitly defined in this example. Result areas for COMPUTE statements may be implicitly expressed or explicitly defined, i.e. DEFINE WORKX 9. A warning message will be issued in the compile phase if the defined work area's length does not correspond with the OP calculated result area length. If the OP calculated work space is longer than the user defined result work

space significant characters of the result will be truncated when the result is moved to the user define work space.

(Note: Result space in a COMPUTE expression is initially set to zero, however, it is not restored to zero when published. The user may use the RESET operator to restore work areas if desired).

The results of all arithmetic expressions will be printed with leading zeroes suppressed and without sign notation unless an EDIT mask has been specified to process the results. Arithmetic processing is performed on integer values only. Resulting fractions from a divide process will not be saved for output or subsequent arithmetic processing. The factors and results associated with all arithmetic calculations should not exceed 15 numeric digits.

3.3.17 Expanded Arithmetic Operations (TOTAL, COUNT, COMPUTE)

In cases where the results of totals, counts, and computations are required, or where the actions of the arithmetic statement rather than the output of the result must be conditioned, another form of arithmetic is required. The three methods are: expansion of the TOTAL or COUNT OF_____statement, the COMPUTE statement, and the COMPUTE card.

Expansion of the TOTAL OF_____Statement

This form allows the user to specify totals and counts in the manner discussed in section 3.3.16 but in addition, the result may be saved for further reference.

The next example illustrates the method by which a total and count can be made and their results divided to produce an average. This average is then printed in position 30. Notice the absence of print positions in the first two statements; output locations could be included if the total and count were to be printed.

```
LINE1 TOTAL OF PERS IN TOTAPER

LINE1 COUNT OF UIC IN CNTUIC
```

```
LINE1 30 TOTAPER DIV CNTUIC
```

The internal procedure for the previous example would be to total the authorized personnel field and move the results to TOTAPER, count the UICs and move the results to CNTUIC, and then perform the division. The internal totaling and counting accumulation areas will be reset to zero after the move has been performed (on each record).

The areas specified for the results can be defined as work areas with define cards.

```
DEFINE TOTAPER 2

DEFINE CNTUIC 4
```

These areas, if not defined in the above manner, will be defined by the output system using the standard lengths, as shown under "Result Areas," section 3.3.16.

These statements specify the results of the operation rather than the action of it. Totals and counts specified in the method mentioned here occur before the processing of the first format specification. The use of OMIT statements can therefore affect the totaling and counting operations.

If the OMIT statement is placed following the FILE card but before the FORMAT card, the total or count will take place on all records that have not been omitted.

If the OMIT statement is placed after a FORMAT statement, the total or count will take place on all records regardless of the conditions specified in the OMIT expression.

Another specification which will affect these operations is the periodic mode specification for a line. The total or count will occur on all records according to the periodic mode indicated for the line which contains the total or count specification.

COMPUTE Statement

The COMPUTE statement form of arithmetic operations can be used in cases where the computed results are needed for later use, for unusual accumulation of totals and counts, and where the performance of the action must be conditioned.

The general form for a total and count is:

LINExx COMPUTE result area EQ TOTAL OF_____

LINExx COMPUTE result area EQ COUNT OF_____

These statements would cause the total or count action to be performed every time the line is printed. The result will be placed in the area designated by the literal name. The next time that the total or count is made, the value will be added to the result area (TOTAL) or the result area will be increased by one (COUNT).

The resultant area may be defined as a work area with a DEFINE card or the user can allow the area to be defined by the system using the standard lengths.

The general form for computations is:

LINExx COMPUTE result area EQ factor, operator, factor, operator, etc.

The factors can be either field names, previously defined literals, or numeric literals.

The computation proceeds from left to right with integer arithmetic. Thus, the factors might have to be scaled to provide the desired accuracy.

Any of the above methods of specifying arithmetic operations may be made conditional by the use of the conditional statements previously described. This will cause the action rather than the output of the result to be conditioned.

COMPUTE Card

The COMPUTE card is needed for those cases where it is not possible or convenient to put the COMPUTE statement on a line, card, or record specification. It could be used if some calculation were performed on a data record and the record conditionally included in the report based upon the results of the computation. Suppose one wished to include all units that possess more than five types of equipment. The specifications to accomplish this would be:

```
FILE TEST360

COMPUTE CNTEQP EQ COUNT OF MEQPT

OMIT IF CNTEQP LT 6
```

MEQPT names are contained as subsets in Periodic Set 1. The COMPUTE statement would count the number of types of equipment (i.e., subsets) and place the count in CNTEQP. This literal could have been defined with a DEFINE card; however, if not previously defined, the literal will be defined using the standard lengths.

Caution should be exercised when using the COMPUTE card with periodic data since the entire set is processed using the COMPUTE statement before the next specification is acted upon.

```
COMPUTE EQPRDY EQ MEORC PLUS MEORN
```

This statement adds the two categories of ready equipment and would result in the literal EQPRDY containing the result from the computation on only the last subset.

The COMPUTE card is also used in conjunction with the generation of matrixes prior to their being output. It is from this level 1 card type that the rows and columns for the specified matrix are determined providing COUNTs and SUMs of the fields. The FOR/AND logical connectors are used similarly as they are used for QUIP matrix generation with the exception that table and subroutine conversion of parameter values is not permitted. Literal values serving as parameters must appear in the FOR/AND statement as they

appear in the file. The basic statements for matrix processing are written in the following manner:

```
COMPUTE MATA EQ COUNT OF UNIT                    X
COMPUTE FOR SERV EQ 'W' 'N' 'J'

COMPUTE MATA EQ SUM OF PERS                      X
COMPUTE FOR SERV EQ 'W' 'N'                      X
COMPUTE AND WC EQ 'PARIS' 'LONDON'
```

The above examples show how the COMPUTE card provides both one- and two-dimensional matrices. (It should be noted that following the FOR/AND logical connectors any of the accepted OP relational operators, except ABSENT and CONTAINS, may be used to qualify the data for the SUMs and COUNTs.) The first example designates one-dimensional matrix with three columns while the second represents a two-dimensional matrix with two columns and two rows. Both matrices MATA and MATB must have been previously defined. The COMPUTE statement will not cause automatic generation of matrix literals.

An asterisk may be used as one of the operands in the literal string, and when so used, it takes on special significance. It provides a universal match condition and causes processing of an appropriate work area for all data elements not associated with an area specifically designated by the list of literals.

3.3.18 Summary Reports (IF COMPLETE)

Summary information is desired in many cases not only involving the entire report, but also on portions of a report. Under normal circumstances, actions conditioned on the presence of a change in a field are processed with the data from the new record present. This is desirable whenever an action on data from a record is conditioned on a change from the previous record.

```
LINE1 IF CNAM CHANGES

LINE1 47 CNAM
```

The field CNAM from the new record would print when CNAM changed.

For summary information applications, it is desirable to condition the action on the basis that a certain type field or record is complete. An example would be to total the number of pieces of equipment possessed for a certain class. The result would be placed on a separate line before the data from the new record is printed (beginning with LINE1 specifications). This is, in essence, a change in the field, but the action desired is different.

The following example assumes that a request from the TEST360 file is ordered by CNTRY and that the total number of weapons located by country is wanted in the report.

```
LINE2 COMPUTE NUMRDY EQ MEORC PLUS MEORN

LINE2 ADD NUMRDY TO TOTRDY

LINE2 50 NUMRDY 6/7 EDIT '##'

LINE3 IF CNTRY COMPLETE

LINE3 60 TOTRDY 6/8 EDIT '###'

LINE3 RESET TOTRDY
```

In this illustration, the two ready categories are added together and the result is added to a literal to achieve a running total of the numbers of readies. Then, when the country is complete, line 3 will be printed containing this total. The literal is then reset to zero to prepare for similar operations in subsequent countries. Line 3 will be printed each time a change occurs in the CNTRY field before any lines containing information about the next country are printed. The line will also be printed at the end of the report.

Any specifications that can be placed on a normal line may be placed on IF ___ COMPLETE lines, except that neither periodic set fields nor periodics may be referenced within a conditional expression. Similarly, variable sets may not be referenced in the BREAKLINE section. All data fields referenced will be from the last record, rather than the record in which the change occurred. Periodic fields or

literals referenced will be only from the last subset in the last record.

Note that the specification differs from the IF_____CHANGES conditional statement in that the latter would cause printing of a line on the first occurrence of a country while the IF_____COMPLETE statement would cause printing after the last occurrence.

IF_____COMPLETE conditionals may also be specified at field or action level; but only if the line on which it occurs is also conditioned by an IF_____COMPLETE. Consider the case of personnel totals by location, country and geopolitical area. One may obtain totals by the various subdivisions in the following manner:

```
LINE3 IF LOC COMPLETE

LINE3 9 'TOTAL FOR'

LINE3 20 LOC

LINE3 26 TOTAL OF PERS

LINE3 30 CNTRY IF CNTRY IS COMPLETE

LINE3 36 TOTAL OF PERS IF CNTRY COMPLETE

LINE3 40 GEPOL IF GEPOL COMPLETE

LINE3 46 TOTAL OF PERS IF GEPOL COMPLETE
```

The above line would be printed as follows if only LOC changed:

```
TOTAL FOR PENTAGON 25026
```

If both country and location changed, the line would be:

```
TOTAL FOR PENTAGON 25026 UNITED STATES 9612436
```

and if location, country and geopolitical area changed:

```
TOTAL FOR PENTAGON 25026 UNITED STATES 9612436
```

NORTH AMERICA 190013245

Whenever a format control statement includes the conditional IF_____COMPLETE, the statement must also include BETWEEN RECORDS. For example:

```
SPACE 3 BETWEEN RECORDS IF CNAM COMPLETE
```

The calculation of periodic values in conjunction with IF COMPLETE logic is not accepted in NIPS 360 FFS. It was accepted in NIPS 1410 FFS but not supported. RITs trying to use this logic will not structure and must be rewritten.

An example of this <u>illegal</u> statement is:

```
LINE4 IF SORT 4/4 COMPLETE
LINE4 ADD PERFLD TO PERLIT
```

where PERFLD is a field from a periodic set and PERLIT is a previously defined numeric literal (fixed or periodic).

Note: IF COMPLETE lines may not be intermingled with other line specifications not intended to have IF COMPLETE logic applied. The first encounter of an IF COMPLETE line sets the mode for all other line specifications and action statements within that format.

3.3.19 Final Lines (FINALLINEn)

It is possible to specify final lines that will be printed only at the conclusion of the report. These lines cannot contain data from a record unless it has been stored in a literal at some previous time. The normal use for this type of line would be for such operations as totals and counts or other types of summary data. After the last record has been processed, any final lines will be printed and then the trailer lines will be printed.

```
FINALLINE1 10 'TOTAL'

FINALLINE1 17 TOTAL OF PERS
```

The above specifications would place the total of all authorized personnel for all processed records into position 17 preceded by the word TOTAL. The conditioning of these

lines follows the same restrictions as header lines in that the parameter of the conditional statement cannot be a file field. Note: Summary information may be obtained on portions of the report.

3.3.20 EXTRACT Statement

While processing data records from one file, it may be desirable to retrieve data values from a second file (Interfile Output). This may be done through the use of the EXTRACT statement. This statement may only be specified within the first file of a multifile FIT. The first file will be known as the primary file, while other referenced files are secondary.

The general form of the EXTRACT statement is as follows:

	FIELD NAME SORTKEY	SUBRT		
EXTRACT	LITERAL	TABLE	FILE NAME	IF...
	IFOGRPn	m/n		

In the above examples of statement syntax, "FIELD NAME" is any field specified in the FFT of the primary file, the file containing pointers or Record IDs for secondary files. The "SORTKEY" is the result of ordering the RASP answer set, while a "LITERAL" is any defined alpha or numeric literal.

"IFOGRPn" is a special case of a group name within the FFT of the primary file where n is any alphanumeric character. This field was designed especially for the Interfile Output capability. Fields designated as IFOGRPn must have at least two fields. The first contains the name of a secondary file or an indicator code which is converted to the secondary file name by a subroutine or table specified in the FFT and the second contains a pointer to the desired data record or group of records in the secondary file.

As shown, subroutine conversion and partial field notation may be used on the EXTRACT statement. The exception to this is IFOGRPn which may not make use of these functions. The two required fields of IFOGRPn may have subroutine conversion specified in the FFT. Unlike the manner in which other groups are processed, automatic

conversion of these fields will take place when IFOGRPn is referenced on the EXTRACT statement.

On all forms of the EXTRACT statement, a file name is required except when IFOGRPn is used. The file name is required to show the secondary file from which data is requested. As was previously stated, the first field of the IFOGRPn must contain a file identifier. If a file name should be referenced when specifying IFOGRPn, the EXTRACT operation will take place only when the file identified by the first field of the group matches the file name specified on the EXTRACT statement.

The file name specified on the EXTRACT statement, and the file identifier in the first field of the IFOGRPn is a maximum of seven characters in length. The DD statements which correspond to the secondary files being referenced may contain a qualified data set name up to 44 characters in length. The last segment of the qualified data set name must match the file name designated on the EXTRACT statement.

The standard OP conditional statement may be used to determine when the EXTRACT statement should be executed.

When the EXTRACT statement is encountered in RIT, processing of the primary file is interrupted and the field specified is used as the key for record retrieval from the secondary file.

Example:

```
EXTRACT   UIC   FILEZ   IF   UIC  NE ' '.
```

In the above example, if the condition is met, the contents of the field UIC will be used as a Record ID in FILEZ. When a data record is found with that key, the data record is processed by the specifications written for FILEZ and control is returned to the statement following the EXTRACT statement in the primary file specifications. If no record is found, control returns immediately to the primary file processing.

Several data records might be retrieved by using a field that is shorter in length than the key length of the

secondary file. Assuming the Record ID of the secondary file is six characters in length and using the following:

```
DEFINE BUCKET '      ' a 6-character alpha literal
     :
     :
MOVE UIC TO BUCKET
EXTRACT BUCKET 1/1 FILEC IF UIC 1/1 EQ 'J'
```

All records with a record key beginning with a 'J' will be retrieved from FILEC and processed. When a field specified on the EXTRACT statement is longer than the Record ID of the secondary file, it will be truncated to the secondary file key length.

Data fields containing record keys to secondary files may be from the fixed or periodic sets. (They may not be variable.) When operating with periodic sets in the primary file, OP will operate in the "ALL" mode for SOURCE DIRECT and in the "SCAN" mode for SOURCE RETRIEVAL runs. Periodics from the secondary files are always in the "ALL" mode.

The processing of periodic fields on the EXTRACT statement will occur as follows. The first subset will be accessed to obtain a record key. Once found, the data record(s) is processed by the secondary file RIT specifications. When the data record(s) have been completely processed, control is returned to the primary file where the second subset is accessed. When all subsets of the primary file have been exhausted, processing of the primary file continues with the next RIT specification.

A Report Instruction Table that contains an EXTRACT operator may be executed in Source Direct mode. In this case the input data sets to OP are primary and secondary file. The primary file may be sequentially or index sequentially organized. The secondary file(s) must be index sequentially organized. Specifications for the primary and secondary files follow the rules for merged file RITs discussed in section 5.1.3 and 5.1.4 with the following exception. The primary file can be designed as one of the secondary files. When processing one record in the primary file, the user can extract data from other records in the primary file. The primary file must have the first set of RIT specifications for primary processing and another set of

specifications that will be invoked only in response to an EXTRACT statement.

The RIT with the EXTRACT operator may also be excuted in Source Retrieval mode. The input to OP will be an answer (QRT/QDF) produced by RASP. The input to RASP should be specified as the primary file in the RIT. As with Source Direct, the primary file may also be designated as a secondary file for use with EXTRACT.

The QRT/QDF can be from a single file or merged file retrieval. In either case the user should exercise caution in addressing the SORTKEY in secondary file specifications as result of an EXTRACT statement with an IF COMPLETE condition. Only the current SORTKEY (the one following the one on which the IF COMPLETE condition was met) is available to the secondary file specifications. This is not the same SORTKEY the system saved for use in the breakline (IF COMPLETE) block for the primary file. If this presents a problem, SORTKEY data should be moved to user specified literals ahead of the EXTRACT statement.

Use of EXTRACT with a merged file retrieval requires an understanding of how the system functions. Each QRT record (SORTKEY) is flagged with the name of the file from which it was retrieved. These flags indicate which primary or secondary set of RIT specifications will process the QRT/QDF record. Since EXTRACT statements are permitted only in the primary file specifications, they will only be executed when a QRT record pointing to the primary file is encountered. A QRT record pointing to a secondary file will bypass the primary file specifications and execute the appropriate secondary file specifications using the QRT/QDF records as input. If the same secondary file is referred to in an EXTRACT statement, the same secondary file specifications will be executed with the record(s) specified in the EXTRACT statement from the secondary data file itself, not the QRT/QDF, as input. The secondary file specifications must be designed to properly process either input. When an EXTRACT statement calls for the secondary file with the same name as the primary file, the second set of specifications with that file name will be executed with the record(s) from the data file as input. Only the QRT record (SORTKEY) pointing to the primary file is available to secondary file processing in response to an EXTRACT statement.

3.3.21 GOTO Statement

A means of altering the order of RIT execution is provided through the use of the GOTO statement and its associated TAGn statement (n is a 1- to 4-digit number). Normally, OP operates in a line-by-line manner as it processes each data record. The GOTO statement enables the OP user to control the order of execution of the RIT.

Essentially, the GOTO statement provides the user with a means of branching. The general form of the statement is as follows:

```
GOTO TAGn   [IF...]
GOTO NEXT RECORD  [IF...]
TAGn
```

In the above, TAGn is the label to which control should be passed. The use of the IF conditional statement is optional. The "GOTO NEXT RECORD" statement is a special usage of the GOTO statement. When this statement is encountered, processing of the current data record is terminated and the next data record available for processing is obtained. This statement, therefore, allows the user to exit from the RIT without executing additional RIT specifications.

The GOTO statement is allowed in all sections of an RIT where LINEs may appear. This means that it may not appear between HEADERS, TRAILERS, OVERFLOWS, etc. There are other restrictions. A GOTO may not be placed within a LINE block. A GOTO may not branch between formats in a multiformat RIT, nor may it branch between files in multifile runs. Within a single format, a GOTO may not traverse the boundaries between regular RIT specifications and those within the breaklines (IF COMPLETE) section.

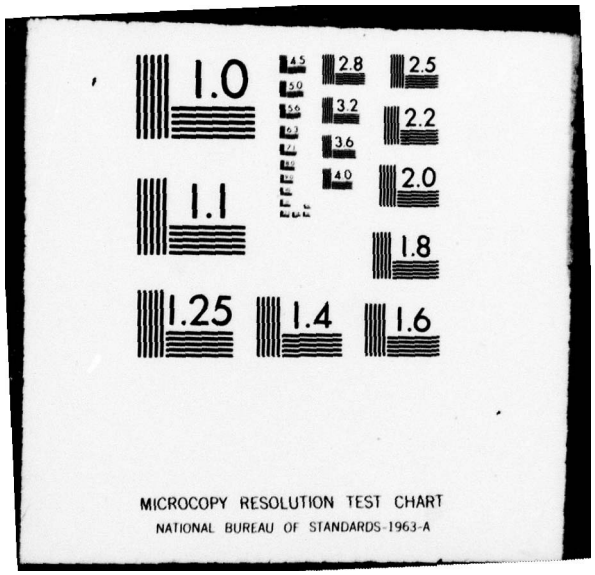
3.3.22 PERFORM Statement

Similar to the GOTO statement, the PERFORM statement enables the RIT to branch on command to a single LINE, CARD or RECORD, to execute the same, and return to the next sequential OP statement following the PERFORM.

AD-A063 430 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON D C F/G 9/2
NMCS INFORMATION PROCESSING SYSTEM 360 FORMATTED FILE SYSTEM (N--ETC(U)
SEP 78 C K HILL
UNCLASSIFIED CCTC-CSM-UM-15-78-VOL-5 SBIE-AD-E100 130 NL

2 OF 2
AD
A063430

END
DATE
FILMED
3-79
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
7.1
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

The execution of the PERFORM statement may be conditioned using the standard OP conditional statement. It is allowed in the standard RIT area, that is, in those sections where LINEs may appear. The execution of this branching capability is not allowed across the boundaries between regular RIT specifications and those within the breaklines (IF COMPLETE) section, nor is it allowed between files of a multifile RIT or between formats of a multi-format RIT.

The following are examples of the PERFORM statement:

```
PERFORM LINEX
PERFORM LINEX IF...
```

In the above, LINEX is the operand of the PERFORM statement, and it designates the particular LINE, CARD or RECORD to be executed or performed. IF is the standard OP conditional statement and its usage is optional. The PERFORM statement, like GOTO, provides the user with a means of branching. The main differences are that the PERFORM returns and it refers only to individual LINEs, CARDs and RECORDs.

3.3.23 Self-Defining Literals

A self-defining literal consists of one to 52 characters enclosed by apostrophes. All letters, decimal digits and special characters, except single quotes, may be used in the literal. Following examples demonstrate the use of self-defining literals.

```
HEADER1 79 '***ANNUAL REPORT - 1974***'
LINE3 120 '$'
DEFINE LIT1 'JUNE && JULY'
```

NOTE: In the last example a double ampersand is used to represent a single literal ampersand for output. Because of the use of the ampersand as a special character in assembler language the rule for two ampersands must be observed when using this character in a self-defining literal.

3.4 Specifications for Punched Output

Although the specifications for punched output are similar to printed output, a number of differences are indicated in the following paragraphs.

3.4.1 Specification of Format (FORMAT)

The specifications for punched output are given separately from other formats and are preceded by a FORMAT card.

```
FORMAT PUNCH SIZE 80
```

The size specification (keyword SIZE) can normally be omitted and will be assumed to be 80 columns. Other sizes can be specified if desired. The output program will check the specified output locations against the size specification to see that the size is not exceeded.

3.4.2 Card Layout (CARDn)

The specifications for card layout containing data fields, literal, etc., are given on input cards with CARD as the card type.

```
CARD1 47 STATE
```

The contents of these specifications are the same as mentioned in the discussion of line specifications under printed output.

```
                    card level conditional statement

                    47 data field

        CARDn       65 literal

                    78 'ACTUAL LABEL'

                    action statement
```

When using the CARD format, numerically defined fields are not automatically edited. They should therefore be edited by the user with an all blank edit mask. This will insure the proper punching of these fields.

Areas, i.e. literals, referenced in the preceding formats can be referenced also in punched formats provided the areas are not cleared before they are referenced in the succeeding format. Thus, the results of totals, counts, and calculations, as specified previously, can be used only once.

3.4.3 Title and Final Cards (TITLECARD, FINALCARD)

Title cards can be specified with the use of the TITLECARD specification. The same criteria must be followed for these cards as with TITLELINE specifications. Similarly, final cards can be specified with FINALCARD specifications.

The discussion relative to format control is not applicable to punched output.

3.4.4 Example

The following example of two-position alpha sequencing on card output is shown for its direct application, as well as for variations that may have other applications.

```
DEFINE PER CNTR1 'ØØ'

CARD1 ADD Ø1 to CNTR1

CARD1 80 CNTR1 SUBRT TABlS
```

Here the counter (CNTR1) will store the count as a numeric field which can then be converted to another form using the subroutine TABlS.

3.5 Specifications for Tape Output

The format specifications for magnetic tape reports also follow the same rules as for printed reports.

3.5.1 Format Control (FORMAT)

The basic FORMAT control criteria are specified on the FORMAT card with keywords in the following general format.

```
FORMAT TAPE  NAME SAMPLOUT  RECORD 80  BLOCK 848
```

Keyword	Function
NAME	Eight-character name to be used as the data set name for the created data set.
RECORD	Record size (up to three digits). The Output Processor produces variable length blocked records; thus the record size indicated here could not be used to specify the actual record length. It is used to specify a maximum record length allowable. The 4-character record count is not included. If nothing is specified, the maximum size will be 200 characters.
BLOCK	Block size (up to 1004). If not specified, the block size will be set to 996. Since the record size given does not include the record character count (four bytes), each variable length record will actually be four bytes longer than the record specified. To figure a minimum block size, take the longest record length, add four for the record character count and add four for the block count. If all records are the same size (for example 80), the block size for a blocking factor of 10 would be: [(80+4) * 10] + 4 = 844 + 4 = 848

Note: The record size of any one record (RECORD14 for example) is the highest record position referenced. For example RECORD14 402-FLDX would generate a 402-byte record preceded by a record character count (four bytes) containing a count of 406. The maximum record size is 999.

3.5.2 Record Card (RECORDn)

The specifications for each record are given on record input cards and follow the same rules stated in the discussion of printed reports.

```
                    record level conditional statement

                    47 data

      RECORD n      65 liberal

                    78 'ACTUAL LABEL'

                    action statement
```

When using the RECORD format, numerically defined fields are not automatically edited. They should therefore be edited by the user with an all blank edit mask. This will insure the proper recording of these numeric fields.

The size of each output record is determined by the largest output position specified for that record plus the four positions used to specify the record character count.

In addition, TITLERECORDS and FINALRECORDS may also be specified.

Section 4

REPORT SPECIFICATIONS SUMMARY

This section contains a summary of the specifications used in preparing a Report Instruction Table (RIT). The summary presents in a consolidated manner all of the specifications which are valid at each level of a RIT statement; it will be useful as a quick reference for RIT preparation and will facilitate the use of error diagnostics.

A RIT may contain up to five levels of specifications. The first level is always the statement type. The following example illustrates a 5-level specification.

```
LINE1 2Ø FLDX SUBRT CONVRT IF FLDX GT 100
```

where:
- LINE1 is a level 1 expression
- 2Ø is a level 2 expression
- FLDX is a level 3 expression
- SUBRT CONVRT is a level 4 expression
- IF FLDX GT 100 is a level 5 expression

The following pages contain two tables which summarize report specifications. Table 1 lists the valid expressions at each of the five levels. Table 2 further refines these valid expressions by presenting those expressions valid at each level by statement type.

4.1 Valid Expressions by Level

The following table lists the expressions which are considered valid at each level by the OP edit programs.

Table 1

REPORT SPECIFICATION SUMMARY
(Valid Expression by Level)
(Page 1 of 6)

Level 1 Expressions

Control, Definition, Action Statements

FILE	Precedes FORMAT for single file
FORMAT	Precedes FILE for merge file
DEFINE	May appear anywhere before being used
OMIT	After FILE (Note)
STOP	After FILE (Note)
MOVE	Anywhere in RIT specification
COMPUTE	Anywhere in RIT specification
ADD	Anywhere in RIT specification
SUB	Anywhere in RIT specification
EXTRACT	In primary file of multifile run
GOTO	Anywhere in RIT specification
TAGn	Anywhere in RIT specification
PERFORM	Anywhere in RIT specification
RESET	Anywhere except within LABEL-LINE block
END	Last card in RIT specification

Note: IF OMIT and/or STOP appears before FORMAT, they affect all FORMATS in the RIT specifications; if after FORMAT, only this format is conditioned.

Table 1

REPORT SPECIFICATION SUMMARY
(Valid Expression by Level)
(Page 2 of 6)

Level 1 Expressions (continued)

Print Statements (Print statements should appear in this sequence when used in the RIT specification.)

- HEADER
- TITLELINE
- OVERFLOW
- LABEL
- LINE
- LINE (IF COMPLETE)
- FINALLINE
- TRAILER
- SPACE — Adjacent to associated print line
- SKIP TO — Adjacent to associated print line
- EJECT — Adjacent to associated print line

Card Statements (Card statements should be in this sequence when used in the punch specification.)

- TITLECARD
- CARD
- CARD (IF COMPLETE)
- FINALCARD

Record Statements (Record statements should be in this sequenc when used in the record specification.)

- TITLERECORD
- RECORD
- RECORD (IF COMPLETE)
- FINALRECORD

Table 1

REPORT SPECIFICATION SUMMARY
(Valid Expression by Level)
(Page 3 of 6)

Level 2 Expressions

IF (Conditional)	Line level condition
XXX	Print, card or record position
XXX XXX XXX	Multiple print positions
XXX-XXX	Variable Field/Set print positions
TOTAL OF _ IN _	
COUNT OF _ IN _	
COMPUTE _ EQ _	
COMPUTE _ EQ TOTAL OF _	
COMPUTE _ EQ COUNT OF _	
ADD _ TO _	
SUB _ FROM _	
MOVE _ TO _	
RESET _	
RESET _ TO _	
ALL	Mode specification for Set Field
SCAN	Mode specification for Set Field
FIRST	Mode specification for Set Field
LAST	Mode specification for Set Field
FIELDNAME	EXTRACT only
FIELDNAME (Partial)	EXTRACT only
SORTKEY	EXTRACT only
SORTKEY (Partial)	EXTRACT only
LITERAL NAME	EXTRACT only
LITERAL NAME (Partial)	EXTRACT only

Table 1

REPORT SPECIFICATION SUMMARY
(Valid Expression by Level)
(Page 4 of 6)

Level 2 Expressions (continued)

IFOGRPn	EXTRACT only
TAGn	GOTO only
NEXT RECORD	GOTO only
LINEn	PERFORM only
RECORDn	PERFORM only
CARDn	PERFORM only

Level 3 Expressions

Fieldname
Fieldname (partial)

Table 1

REPORT SPECIFICATION SUMMARY
(Valid Expression by Level)
(Page 5 of 6)

Level 3 Expressions (Continued)

Self-defining Literal	
Literal Name	
Literal Name (partial)	
System Label	
TOTAL OF _	
COUNT OF _	
TOTAL OF _ IN _	
COUNT OF _ IN _	
Arithmetic expression	
IF (Conditional)	
SUBRT X	Use SUBRT X (EXTRACT only)
TABLE X	Use TABLE X (EXTRACT only)
#SUBNAME#	Use subroutine named 'SUBNAME' (EXTRACT only)
##	Override subroutine (EXTRACT only)
FILENAME	(EXTRACT only)

Level 4 Expressions

TABLE	May be automatic (Note)
SUBRT	May be automatic (Note)
#Subname#	Use subroutine named 'Subname'
EDIT	Use FFT Edit
EDIT ' '	Use Edit
Edit Word Name	User-defined edit word
##	Override Subroutine
''	Override Edit
## ''	Override Both
SUBRT EDIT ' '	Use FFT Subroutine and Edit
SUBRT X	Use Subroutine X
SUBRT X EDIT	Use Subroutine X and FFT Edit
TABLE EDIT ' '	Use FFT Table and Edit
TABLE X	Use Table X
TABLE X EDIT	Use Table X and FFT Edit

Table 1

REPORT SPECIFICATION SUMMARY
(Valid Expression by Level)
(Page 6 of 6)

IF (Conditional)
FILENAME (EXTRACT only)

Note: Subroutines, Tables and Edits are not automatic on Card and Record specifications.

Level 5 Expressions

IF (Conditional)

4.2 Valid Expressions at Each Level by Statement Type

The following table is a summary of valid expressions, organized by statement type.

Table 2

REPORT SPECIFICATIONS SUMMARY
(Valid Expression by Level - FILE and FORMAT)
(Page 1 of 18)

LEVEL 1		POSSIBLE ENTRIES
FILE	XXXXXXX	XXXXXXX = The name of the file (1 to 7 characters)
FORMAT	PRINT PRINT LINES AA PRINT SPACE B PRINT SIZE CCC PRINT LINES AA SPACE B SIZE CCC PUNCH PUNCH SIZE CC TAPE NAME DDDDDDDD RETAIN EEE RECORD FFF BLOCK GGGG	
OMIT IF (Conditional)		If placed after the FILE card and before the FORMAT, will OMIT all formats. (FILE CONTROL SECTION). If placed after the FORMAT card will omit only that format.
STOP IF (Conditional)		Should immediately follow FILE card for the file it pertains to.

COMMENTS		DEFAULTS
AA	= Number of body lines per page	50
B	= Normal spacing	1
CCC	= Number of output positions required	Print - 132, Card - 80
DDDDDDDD	= Eight-character data set name of the tape being written	OPTAPE
EEE	= Retention period in days	10
FFF	= Number of characters in longest record (for error checking only)	200
GGGG	= Block size in characters (maximum of 9999)	996

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expression by Level - DEFINE)
(Page 2 of 21)

LEVEL 1	POSSIBLE ENTRIES	FIELD TYPE
DEFINE	NAME '___'	Alphabetic, 1 to 52 chars.
	NAME LLL	Decimal, 1 to 999 digits
	NAME B LLL	Binary, 1 to 10 digits
	PER NAME '___'	Alphabetic
	PERIODIC NAME '_____'	Alphabetic
	PER NAME LLL	Decimal
	PERIODIC NAME LLL	Decimal
	SET(b) NAME '_____'	Alphabetic
	SET(b) NAME LLL	Decimal
	PER NAME B LLL	Binary
	PERIODIC NAME B LLL	Binary
	SET(b) NAME B LLL	Binary
	SET(b) NAME WORK(N) LLL	Binary (default mode), 1 to 10 digits
	SET(b) NAME WORK(N) LLL A	Alphabetic, 1 to 4 chars.
	SET(b) NAME WORK(N) LLL B	Binary, 1 to 10 digits
	SET(b) NAME WORK(N) LLL C	Coordinate, 5 to 8 digits
	SET(b) NAME WORK(N) LLL D	Decimal, 1 to 4 digits

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expression by Level - DEFINE)
(Page 3 of 21)

LEVEL 1	POSSIBLE ENTRIES			FIELD TYPE
	EDIT	NAME	'___'	Alphabetic 1 to 26 characters
	NAME	(X)	LLL	Decimal
	NAME	(X)	B LLL	Binary
	NAME	(X,Y)	LLL	Decimal
	NAME	(X,Y)	B LLL	Binary

COMMENTS

NAME = 1-7 character literal name. Must begin with an alphabetic character.

LLL = The 1-3 digit literal length.

(b) = 1-3 digit set number, range 1 - 255.

WORK = 4-byte fullword work areas generated in the retriever and passed to OP by set number. They may be defined as binary fullwords, internal form coordinate, 1 - 4 byte alphabetic or decimal fields.

(N) = 1-digit work area number, range 1-5.

(X) = The 1-3 digit occurrence number.

(X,Y) = The 1-3 digit row and column numbers.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - ADD, SUB, MOVE, COMPUTE)
(Page 4 of 21)

Level 1	Level 2
ADD ___ TO ___	IF (Conditional)
SUB___ FROM___	
MOVE ___ TO ___	
MOVE ___ SUBRT X TO ___	
MOVE ___ TABLE X TO ___	
COMPUTE ___ EQ COUNT OF ___	
COMPUTE____EQ SUM OF_____	
COMPUTE ___ EQ TOTAL OF ___	
COMPUTE ___ EQ ___ MUL___ DIV ___ ADD ___ SUB ___	
COMPUTE ___ = ___ * ___ / ___ + ___ - ___ .	

Note: The above action statements can also appear at levels 2 and 3 of other statements. The exception is when a matrix is being generated making use of SUM OF or COUNT OF. Matrix generation statements are only valid at level 1.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - RESET, SPACE, SKIP, EJECT)
(Page 5 of 21)

Level 1	POSSIBLE ENTRIES Level 2	
RESET	Literal Name	IF (Conditional)
	System Label	
	Literal Name TO ___	
	System Label TO ___	
	Note: Areas are either reset to zeros or to the literal specified with a true sign.	
SPACE	n	IF (Conditional)
	n BEFORE (card type)	
	n BETWEEN RECORDS	
	n = Number of spaces required	
EJECT	BEFORE (card type)	IF (Conditional)
	BETWEEN RECORDS	
SKIP TO	n	IF (Conditional)
	n BEFORE (card type)	
	n BETWEEN RECORDS	
	n = Carriage channel to SKIP TO	

Note: If a format statement and a conditional statement appear on the same line for SPACE, SKIP TO, or EJECT, then the conditional statement must be last.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - HEADER, TRAILER, TITLELINE, TITLECARD, TITLERECORD)
(Page 6 of 21)

POSSIBLE ENTRIES

Level 1	Level 2	Level 3	Level 4
HEADER(nnn)	IF (Conditional)		
TRAILER(nnn)	XXX		
TITLELINE(nnn)			
TITLECARD(nnn)			
TITLERECORD(nnn)			
		XXX '_____'	IF (Conditional)
		System Label	
		Literal	
		SORTKEY X/Y (Note)	
		SORT X/Y (Note)	

COMMENTS

nnn = 1-3 digit line level (Alpha levels may be used but should not be mixed with nonalpha levels)

XXX = 1-3 digit print position (This position is the low-order (rightmost) position.)

X/Y = partial field notation

Note: The partial field notation for the sort key must not include positions 1-3 of the sort key.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - LABEL)
(Page 7 of 21)

POSSIBLE ENTRIES			
Level 1	Level 2		
LABELnnn	IF (Conditional)		
LABELnnna	XXX		
	PERIODIC		
	PER		
	SET(b)		
	Level 3		Level 4
	XXX	'_____'	IF (Conditional)
		System Label	
		Literal	
		SORTKEY X/Y (Note)	
		SORT X/Y (Note)	

Note: The partial field notation for the sort key must not include positions 1-3 of the sort key.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - LABEL)
(Page 8 of 21)

POSSIBLE ENTRIES

COMMENTS

nnn = 1 - 3 digit line level
nnna = Alpha line level which is a 1 - 3 digit number followed by one alphabetic character
XXX = 1 - 3 digit print position (This is the low-order (right-most) position.)
(b) = 1 - 3 digit set number
X/Y = partial field notation

Notes: Label lines with alphabetic level designation should have only one alpha level conditional expression per group of labels. The conditional should precede the group, and should contain the alpha character which starts the group.

The partial field notation for the sort key must not include positions 1-3 of the sort key.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - LINE, CARD, RECORD, OVERFLOW)
(Page 9 of 21)

POSSIBLE ENTRIES		
Level 1	**Level 2**	
LINEnnn	IF (Conditional)	
LINEnnna	XXX XXX XXX	
CARDnnn	XXX	
CARDnnna		
RECORDnnn	Action Expression (Group 1)	
RECORDnnna		
OVERFLOWnnn	XXX/XXX Variable Field	
OVERFLOWnnna	ALL	
	ALL SET(b)	
	SCAN	
	SCAN SET(b)	
	FIRST m	m is the count for FIRST and LAST. Range is 1 - 255.
	FIRST m SET(b)	
	LAST m	
	LAST m SET(b)	

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - LINE, CARD, RECORD, OVERFLOW)
(Page 10 of 21)

POSSIBLE ENTRIES			
Level 1	Level 2	Level 3	
	XXX	Field Name	
	XXX	Literal	
	XXX	System Label	
			Level 4
	XXX	'___'	PERIODIC
			PER
			SET(b)
	XXX	Field Name	EDIT
		Action Expression (Group 2)	TABLE use FFT entry
		Literal	SUBRT use FFT entry
			EDIT 'edit word'
			SUBRT NAME
			TABLE NAME
			#SUB/TAB#
			## (override FFT Subrt)

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - LINE, CARD, RECORD, OVERFLOW)
(Page 11 of 21)

POSSIBLE ENTRIES
Level 4 (Cont'd)
'' (override FFT Edit)
IF (Conditional)

Level 5

IF (Conditional) (Applies when a TABLE/SUBRT/EDIT expression is at level 4)

COMMENTS

nnn = 1 - 3 digit line level
nnna = Alpha line level which is a 1 - 3 digit number followed by one alphabetic character.
XXX = 1 - 3 digit print position
(b) = 1 - 3 digit set number
m = 1 - 3 digit count for FIRST or LAST

Note: OVERFLOW is like FINALLINE if the RIT is a merge file RIT. No file fields are allowed.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - FINALLINE, FINALCARD, FINALRECORD)
(Page 12 of 21)

POSSIBLE ENTRIES			
Level 1	Level 2	Level 3	Level 4
FINALLINEnnn	IF (Conditional)		
FINNALLINEnnna	XXX	'__'	IF (Conditional)
	XXX XXX ...	Array or Matrix	
FINALCARDnnn		System Label	
FINALCARDnnna		Literal	
FINALRECORDnnn		SORTKEY X/Y (Note)	
FINALRECORDnnna		SORT X/Y (Note)	
		Action Expression (Group 2) - only difference between HEADERS, TRAILERS, TITLELINES, TITLECARDS	
	Action Expression (Group 3)	IF (Conditional)	

Note: The partial field notation for the sort key must not include positions 1-3 of the sort key.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - FINALLINE, FINALCARD, FINALRECORD)
(Page 13 of 21)

POSSIBLE ENTRIES

COMMENTS

nnn = 1 - 3 digit line level
nnna = Alpha line level which is a 1 - 3 digit number followed by one alphabetic character
XXX = 1 - 3 digit print position
X/Y = partial field notation

Notes: No file fields are allowed except in certain action expressions, namely COUNT OF___, TOTAL OF___, TOTAL OF___ IN___, COUNT OF___IN___. In merge file reports, file fields are not allowed in FINALLINES, FINALCARDS, or FINALRECORDS.

The partial field notation for the sort key must not include positions 1-3 of the sort key.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - IF (Conditional))
(Page 14 of 21)

POSSIBLE CONDITIONAL EXPRESSIONS		
Level 2,3,4, or 5		
IF PARAMETER	OPERATOR	PARAMETER
	(NOT) GREATER	
	" LATER	
	" AFTER	GT (GREATER THAN)
	" GT	
	" LESS	
	" EARLIER	
	" BEFORE	LT (LESS THAN)
	" LT	
	" EQUAL	
	" EQUALS	
	" EQUALING	EQ (EQUAL)
	" EQ	
	" =	
	" CHANGE	CHANGES
	" CHANGES	
	" CH	(No parameter after)
	" COMPLETE	
	" NGT	NOT GREATER THAN
	" NLT	NOT LESS THAN
	" NCH	NO CHANGE
	" NEQ	NOT EQUAL
	" NE	NOT EQUAL——
	" NLTE	NOT LESS THAN OR EQUAL

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - IF (Conditional))
(Page 15 of 21)

POSSIBLE CONDITIONAL EXPRESSIONS

	OPERATOR	PARAMETER
"	LTE	LESS THAN OR EQUAL
"	LE	LESS THAN OR EQUAL
"	NGTE	NOT GREATER THAN OR EQUAL
"	GTE	GREATER THAN OR EQUAL
"	GE	GREATER THAN OR EQUAL
"	CONTAINS	
"	ABSENT	

Note: 'NOT' may be placed in front of any operator.

Parameters may be data fields, literals, system labels, and defined literals. Limitations depend on the literal or field name type and the statement on which the condition appears. Reference should be made to the statement type (level 1) for further restrictions on parameters.

AND and OR may be used to link 'parameter operator parameter' strings.

The following conditional expressions illustrate processing of condition logic.

NON-BOOLEAN - Normal mode of condition code generation -

Expression	Interpretation
IF A AND B	Both A and B must be true for the condition to be met.
IF A OR B	Either A or B must be true for the condition to be met.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - IF (Conditional)
(Page 16 of 21)

POSSIBLE CONDITIONAL EXPRESSIONS	
Expression	**Interpretation**
IF A OR B AND C	C must be true and either A or B must also be true for the condition to be met.
BOOLEAN LOGIC 'BCOL' punched as the last parameter on the CREATE card.	
Expression	**Interpretation**
IF A AND B	Both A and B must be true for the condition to be met.
IF A OR B	Either A or B must be true for the condition to be met.
IF A AND B OR C	Both A and B must be true, or C must be true for the condition to be met.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - Action Statement)
(Page 17 of 21)

POSSIBLE ACTION STATEMENTS BY GROUP

Group 1 Action Expressions

1. TOTAL OF___IN____
2. COUNT OF___IN____
3. COMPUTE___EQ____
4. COMPUTE___EQ TOTAL OF____
5. COMPUTE___EQ COUNT OF____
6. ADD___TO____
7. SUB___FROM____
8. RESET____
9. RESET___TO___
10. MOVE___TO___
11. MOVE___SUBRT NAME TO___
12. MOVE___TABLE NAME TO___
13. COMPUTE____EQ SUM OF_____

Group 2 Action Expressions

1. TOTAL OF___
2. COUNT OF___
3. Arithmetic Expression (X DIV Y, X MUL Y, etc.)
4. TOTAL OF___IN___
5. COUNT OF___IN___

Note: Group 2 action expressions may be followed by SUBRT, TABLE or EDIT if at level 3.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - Action Statement)
(Page 18 of 21)

POSSIBLE ACTION STATEMENTS BY GROUP

Group 3 Action Expressions

1. TOTAL OF___IN___
2. COUNT OF___IN___
3. COMPUTE___EQ___
4. ADD___TO___
5. SUB___FROM___
6. MOVE___TO___
7. RESET___
8. RESET___TO___

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - EXTRACT)
(Page 19 of 21)

POSSIBLE ENTRIES				
Level_1	Level_2	Level_3	Level_4	Level_5
EXTRACT	FIELDNAME	SUBRT NAME	FILENAME	IF(conditional)
	LITERAL	TABLE NAME	IF(conditional)	
	SORTKEY	#SUBNAME#		
	IFOGRPn (Note)	##		
		FIELDNAME		
		IF(conditional)		

Note: n is any alphanumeric character.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level - GOTO, TAG)
(Page 20 of 21)

POSSIBLE ENTRIES		
Level 1	Level 2	Level 3
GOTO	TAGn (Note) NEXT RECORD	IF(conditional)
TAGn (note)		

Note: n is 1 to 4 numeric digits.

Table 2

REPORT SPECIFICATION SUMMARY
(Valid Expressions by Level-PERFORM)
(Page 21 or 21)

POSSIBLE ENTRIES		
Level 1	**Level 2**	**Level 3**
PERFORM	LINEnnn (NOTE) CARDnnn (NOTE) RECORDnnn (Note)	IF (Conditional)

Note: nnn is the 1-3 digit line level

4.3 RIT Code Generation Organization

Publishing a RIT involves execution of the code which is generated by OP from the report specification cards. The sequence of RIT output is determined by the generation and execution of this code in "sections," and within sections by "blocks." Following is a discussion of sections and blocks of code.

4.3.1 RIT Sections of Code

For each data record (or sort key) which is sequentially accessed by OP, the first section of code to be executed is one which handles processing of HEADERS, TRAILERS, OVERFLOWS, TITLELINES, TITLECARDS, TITLERECORDS, FINALLINES, FINALCARDS, and FINALRECORDS. These specification cards do not appear together in a RIT structure deck, but are processed together for execution.

The second section of code, the File Control section, is made up of different types of RIT specifications, depending on where the user has placed the card in his RIT structure deck. For example, in a single file RIT, OMIT, STOP, COMPUTE, RESET, ADD, SUB, and MOVE statements are processed in this section if they appear before the first FORMAT card. If these statements appear after a FORMAT card they are processed within the data section for that FORMAT and would not be executed within the File Control section. In a merge file RIT, the File Control section contains those statements (OMIT, STOP, COMPUTE, RESET, ADD, SUB, and MOVE) which occur after a FILE card and prior to the first LINE statement for the particular file. Examples are shown in section 5, "Run Deck Formats," concerning inclusion of RIT specifications within the File Control section.

The Breaklines section of code is executed next and contains the IF COMPLETE line specifications. This section uses values saved from the previous record and is therefore not executed on the first record.

Following Breaklines, the SAVE section moves values to save literals for LINES, CARDS, and RECORDS. This section also maintains TOTAL OF and COUNT OF work areas.

The PRINT section is then executed. It processes OMITS and STOPS (appearing after FORMAT PRINT card), and LABEL and LINE specifications. At this point, all output for FORMAT PRINT is complete.

The next section is one to process IF COMPLETE card specifications, IF COMPLETE record specifications, CARDS, and RECORDS. If a particular format is not required, the coding for that section will consist of a return to the calling program. Note that in a multiformat RIT, the formats will always be executed in the above-described sequence, regardless of the user's deck sequence.

When more than one file is mentioned in the RIT, there is a repetition of these sections of code for each additional file.

4.3.2 RIT Blocks of Code

Within a section, code is executed in "blocks." A "block" is all statements associated with a LINE. For example "LINE3".

```
e.g.          LINE2  20  ITEM1
              SPACE 2 BEFORE LINE3
              LINE 3 IF ITEM2 COMPLETE
              LINE3A 20 ITEM3
              LINE3B 22 ITEM4
Block         LINE3C 30 TOTAL OF ITEM5
              LINE3C COMPUTE AA EQ BB + CC
              SPACE 2
              EJECT IF BODYLINES LT 10
              LINE4 30 ITEM6
              LINE5 40 AA
```

Action statements, (including ADD, SUB, MOVE, etc.) which are not specified on a line, are also executed in blocks. Blocks are formed from contiguous action statements making reference to the same set.

```
e.g.      ADD   FLDXA  TO    BUCKET          (fixed set)
          ADD   FLD1A   TO   BUCKET
BLOCK     SUB   FLD1B  FROM  BUCKET          (Periodic Set 1)
          MOVE  FLD1C  TO    HOLD1
          MOVE  FLD2A  TO    HOLD2           (Periodic Set 2)
```

A block is cycled through until the referenced set or sets are exhausted. Control is then passed to the next block.

The amount of code generated for a block may not exceed 8,000 bytes. All user-defined and OP-generated literals are also grouped into a data block which is limited to 8,000 bytes.

Section 5

RUN DECK FORMATS

The material in this section illustrates a typical sequence of specification cards which could appear within an OP run deck. Examples, with commentary, are first given for structuring RITs with all combinations of files and formats. The deck sequence is then illustrated for runs in which RITs are executed but not structured. A third section shows the sequence for jobs in which both structuring and execution of the RITs are accomplished.

5.1 OP RIT Structure

At the end of an RIT structure run, a load module RIT (ready for execution) has been placed on a library (temporary or permanent) for each set of RIT specifications (CREATE through END cards). No RITs are executed and no reports are produced in a structure job.

5.1.1 Single File Print Format

In a single-file RIT, data records are processed sequentially from the file if the RIT is executed SOURCE DIRECT. If the RIT is executed SOURCE RETRIEVAL, there is a sequential pass of the QRT with access to each associated data record on the Qualifying Data File (QDF). A single RIT may be structured for both SOURCE DIRECT and SOURCE RETRIEVAL executions (in separate jobs) assuming RIT specifications are the same.

OUTPUT PROCESSOR (OP)

Single File Print Format	Remarks
CREATE RITID=Ritname STORE=PERM-NEW	
FILE	
OMIT, STOP, COMPUTE, RESET, ADD, SUB, MOVE	File Control section
FORMAT PRINT	
OMITS, STOPS	
HEADERS	Top of each page
TITLELINES	Once at beginning
OVERFLOWS	Top of page if not line '1'
LABELS	Print section
LINES	
LINES IF___COMPLETE	Breaklines section
EJECT BETWEEN RECORDS IF COMPLETE	
Note: Breaklines are grouped after all other lines	
FINALLINES	End of report
TRAILERS	End of page
END	

SPACE, SKIP and EJECT adjacent to any printed line.

COMPUTE, ADD, SUB, MOVE, RESET and DEFINE may be anywhere.

GOTO and the associated TAG may be placed anywhere in the Print and Breakline sections. They may not traverse the boundaries between these sections. The same holds true for the PERFORM statement.

5.1.2 Single File, Multiformat

In a multiformat RIT, execution of the formats is always in the order of PRINT, PUNCH, TAPE, regardless of the user's deck sequence. Use of the OMIT and STOP statements with different formats is discussed in section 3.3.7, "Conditional Exclusion of Data Records," section 3.3.7, "Stopping the Report Conditionally," and section 3.3.17, "Expanded Arithmetic Operations (TOTAL, STOP, OMIT, COUNT, COMPUTE)."

Single File, Multiformat	Remarks
CREATE RITID=Ritname STORE=PERM-NEW	
FILE	
OMITS, STOPS	File Control section Entire report
FORMAT PRINT (See Single File Print Format)	
FORMAT PUNCH	
OMITS	This format only
STOPS	For remainder of RIT
TITLECARDS	
CARDS	Card section
CARDS (IF COMPLETE)	Break Cards section
FINALCARDS	

OUTPUT PROCESSOR (OP)

FORMAT TAPE	
OMITS	This format only
STOPS	For remainder of RIT
TITLERECORDS	
RECORDS	Record section
RECORDS (IF COMPLETE)	Break Records section
FINALRECORDS	
END	

COMPUTE, ADD, SUB, MOVE, RESET, and DEFINE may be anywhere.

GOTO and TAG may go in the PRINT, CARD, and RECORD sections as well as their associated Break sections. They may not traverse the Break section boundaries or those between formats. The same holds true with the PERFORM statement.

5.1.3 Merge File, Single Format

A merge file RIT is executed following a RASP step and processes a QDF and QRT. As each entry in the QRT is accessed sequentially, the associated data record in the QDF is processed with those RIT statements for the specific file which contains the current data record. Specifications in the RIT for different files automatically generate separate processing blocks of code so that equal line numbers in different file sections of the RIT will not cause data to be printed on the same physical print line. Work areas defined in the RIT are common to all files. Therefore, data from different files can be saved in work areas and output on one physical print line.

OUTPUT PROCESSOR (OP)

Merge File, Single Format (Retrieval answers only)	Remarks
CREATE RITID=Ritname STORE=PERM-NEW	
FORMAT PRINT	
HEADERS	
TITLELINES	
OVERFLOWS	No file fields
FILE X	
OMITS, STOPS, Action Statements	This format only - File Control section
LINES	
LINES (IF COMPLETE)	If complete lines by file
FILE Y	
OMITS, STOPS, Action Statements	File Control, File Y
LINES	
LINES (IF COMPLETE)	
FINALLINES	File fields not allowed
TRAILERS	
END	

Note: File Control includes all action statements before the first LINEnnn.

The EXTRACT statement may only be specified within the specifications for the first file, FILE X.

5.1.4 Merge File, Multiformat

In a merge file, multiformat RIT, the data record associated with each sort key is processed in the same way discussed in section 5.1.3, "Merge File, Single Format." During processing of each data record, the formats are always executed in the order of PRINT, PUNCH, TAPE.

Merge File, Multiformat (Retrieval answers only)	Remarks
CREATE RITID=Ritname STORE=PERM-NEW	
FORMAT PRINT	
HEADERS	
TITLELINES	
OVERFLOWS	No file fields
FILE 1	
LABELS, LINES	
FILE 2	
LABELS, LINES	
FINALLINES	No file fields
TRAILERS	No file fields
FORMAT PUNCH	
TITLECARDS	
FILE 1	
CARDS	
FILE 2	

```
OUTPUT PROCESSOR (OP)

CARDS

FINALCARDS

FORMAT TAPE

TITLERECORDS

FILE     1

RECORDS

FILE     2

RECORDS

FINALRECORDS

END
```

The EXTRACT statement may only occur within the specifications for FILE1 in the PRINT, CARD, and RECORD formats.

5.2 OP RIT Execution

In one OP Execution run, one RIT will be executed for each PUBLISH card. The execution may be SOURCE DIRECT or SOURCE RETRIEVAL but not both in one execution of OP.

5.2.1 SOURCE DIRECT

Publish Directly from Data Base

```
SOURCE DIRECT

PUBLISH RITID=Ritname  BODYLINES=40 CLASS=UNCLASSIFIED
```

5.2.2 SOURCE RETRIEVAL

Publish from Retrieval Answer Set

```
SOURCE RETRIEVAL

PUBLISH RITID=Ritname  ANSID=nn  CLASS=UNCLASSIFIED
```

5.3 OP RIT Structure and Execution

In one OP run, RITs may be structured and executed. All structure specifications (CREATE cards) come before the SOURCE card, and all execution specifications (PUBLISH card) follow the SOURCE card. The run may be SOURCE DIRECT or SOURCE RETRIEVAL but not both in one execution of OP. Temporary and permanent RITs may be used in the same execution of OP.

5.3.1 Permanent RITs

Store RIT Permanently

```
CREATE RITID=Ritname   STORE=PERM-NEW

(RIT STRUCTURE CARDS)

END

SOURCE DIRECT

PUBLISH RITID=Ritname   DATE=19MAR68 CLASS=UNCLASSIFIED
```

5.3.2 Temporary RITs

Store RIT Temporarily

```
CREATE RITID=Ritname  STORE=TEMP

          (RIT Structure Cards)

END

SOURCE DIRECT

PUBLISH SPECIAL=Ritname CLASS=UNCLASSIFIED
```

5.3.3 Permanent and Temporary RITs in One Job

```
CREATE  RITID=TEMP1  STORE=TEMP
(RIT STRUCTURE CARDS)
END
CREATE  RITID=PERM1  STORE=PERM-NEW
(RIT STRUCTURE CARDS)
END
CREATE  RITID=PERM2  STORE=PERM-NEW
(RIT STRUCTURE CARDS)
END
SOURCE DIRECT
PUBLISH SPECIAL=TEMP1  CLASS=UNCLASSIFIED
PUBLISH RITID=PERM1  DATE='12 FEB 1970'
PUBLISH RITID=PERM5  PARAM=XYZ  CLASS=UNCLASSIFIED
```

In this example, the RIT name TEMP1 is structured, stored on the Temporary Library, and executed. The RIT named PERM1 is structured, stored on the Permanent Library, and executed. The RIT named PERM2 is structured and stored on the Permanent Library but not executed in this run. The RIT named PERM5 is executed from the Permanent Library.

5.4 Checkpoint/Restart

During SAM processing, the user may invoke the OS 360 checkpoint/restart capability to record timed or end-of-volume checkpoints. If checkpoints are needed during ISAM processing, only the timed option is valid. The checkpoint/restart should only be used during long-running jobs using the execute only procedures. (Note that the OS 360 step restart is program independent and is not the topic of this discussion.) A detailed description of the OS 360 checkpoint/restart capability (which is utilized in NIPS) is available to the interested user in IBM Systems Reference Library, Number C28-6708. A detailed description on how to use checkpoint/restart is included in Volume VIII, Job Preparation Manual.

Section 6

SUPPORTING FEATURES

This section describes features outside the OP component which support its operation.

6.1 Operating System/360

The Output Processor operates within the S/360 Operating System. S/360 OS is included as a supporting item only to make the user aware that many of the features of the operating system are used by NIPS or can be used to supplement the capability. Familiarity with OS will be most useful.

6.2 Coordinate Conversion

Coordinate fields are carried in the data file in a special internal format to facilitate various geographic searches. The field is automatically converted to an external format by the Output Processor. The formats are as follows:

Output Length	Result
5	DDMMN/S
6	DDDMME/W
7	DDMMSSN/S
8	DDDMMSSE/W
11	DDMMN/SDDDMME/W
15	DDMMSSN/SDDDMMSSE/W

OUTPUT PROCESSOR (OP)

Coordinate Position

D = DEGREES
M = MINUTES
S = SECONDS

Directional Position

N = NORTH
S = SOUTH
E = EAST
W = WEST

To access a portion of a coordinate field, such as in a comparison for degrees, the field must first be moved to a user-defined work area using the system subroutine UTCORDO. Assume, for example, a coordinate field named LAT (output length 5):

```
DEFINE AREA 5

MOVE LAT SUBRT UTCORDO TO AREA

LINEn ....action.... IF AREA 1/2 GT 30
```

Section 7

SUMMARY OF 1410 NIPS and NIPS 360 FPS CHANGES

This section is included as a conversion aid for NIPS 1410 FPS users who wish to use the control language of NIPS 360 FPS. There are no language changes which would force modifications to an existing RIT before it could be structured. However, certain restrictions will be imposed on the user. These restrictions fall into the category of actions not recommended or warned against in NIPS 1410 FPS, but allowed. In NIPS 360 FPS, they will not be allowed. Also, there are a limited number of expansions which the user might desire to use.

7.1 Alpha Literals

Alpha literals may be enclosed in at (@) signs or quote (') signs. The quote (') and/or ampersand (&) will be illegal within the literal, and will result in a non-structured condition.

Examples:

```
DEFINE LABEL @EXAMPLE ALPHA LITERAL@

     same as

DEFINE LABEL 'EXAMPLE ALPHA LITERAL'

     The following is illegal:

DEFINE LABEL 'ILLEGAL TO USE & OR''
```

7.2 Binary Literals

This new literal allows the user to perform arithmetic-type operations on binary fields without conversion.

Example:

```
DEFINE WORKA B nn          nn=Length of output
```

7.3 Subroutine Table and Edit Designation

A fieldname (printed) with no subroutine/table/edit designation will default to the output subroutine/table/edit functions specified in the FFT. A fieldname followed by the word SUBRT, TABLE, or EDIT will do the same. The FFT designation can be overridden by the fieldname followed by ## (subroutine or table) or "(quote, quote for edit). The output will be in file form or it can be overridden by #SUB/TAB#, SUBRT SUBNAME, TABLE TABNAME, EDIT 'edit mask', or the name of a defined edit mask and the data will be processed using the designated subroutine, table, or edit mask.

Examples:

```
LINE1 20 SERV          )
                       )Uses subroutine or edit word
LINE1 20 SERV SUBRT    )designated in FFT

LINE1 20 SERV ##        Output will be in file form

LINE1 20 SERV #SUBX#     )
                         )  Output will be processed
LINE1 20 SERV SUBRT SUBX)   by subroutine SUBX
```

7.4 MOVE, ADD, SUB Statements

These new statements allow the user to perform these actions without creating dummy (blank) lines and then conditioning them out. In essence, they are on the same level as the LINE, RECORD, CARD, etc.

Examples:

```
    MOVE A TO B  )
                 )
    ADD A TO B   )  Level 1 Action
                 )
    SUB A FROM B )

    MOVE A TO B IF A CHANGES          Level 1 Conditional Action
```

7.5 PARAM System Label

This system label allows the user to introduce information to the RIT at execution time. It is a 60-byte EBCDIC field.

Example:

```
    LINE 1 IF PARAM 1-1 EQUAL 'A'  Test the first
                                   character of PARAM for A
```

7.6 Partial Field Designation

The HOP and LOP on partial field designation can be separated with either a - (dash) or /(slash).

Example:

```
    LINE 1 IF SORT 5-6 CHANGES)
                               ) Partial field designation
    LINE 1 IF SORT 5/6 CHANGES) is the same
```

7.7 OP Control Cards

7.7.1 Source Off-line

Not valid. Same results can be obtained through S/360 OS and the utility programs.

7.7.2 Create

The CREATE card has been expanded to allow the user some control over the disposition of the RIT. Note, however, that "Special" RITs must now have a CREATE card.

Examples:

a.	CREATE RITAR	Will result in the RIT RITAR being structured and stored on the permanent library as a change; the change and the structure will not be listed.
b.	CREATE RITID=RITNAME STORE=PERM-OLD	Structure the RIT RITNAME. The store and list options are the same as in example 1.
c.	CREATE RITID=RITNAME STORE=PERM-NEW	Structure the RIT RITNAME. The newly structured RIT is added to the permanent library.
d.	CREATE RITID=RITNAME STORE=TEMP	Disposition to be used for specials. If TEMP is designated, the RIT will be added to the temporary library and deleted on completion of the job.
e.	CREATE RITID=RITNAME DEBUG	Designates a debug pass through the edit and translator to check for specification errors. No RIT will be structured or stored.
f.	CREATE RITID=RITNAME STORE=PERM-NEW BOOL	Example of the use of the Boolean logic option.

7.7.3 Off-line

The OFFLINE control card is not valid. If the user desires to obtain offline print or punch, he may do so by overriding the proper DD cards when generating the output and using the standard S/360 OS print punch utility to punch or list the output.

7.7.4 Publish Special

The PUBLISH SPECIAL card is invalid.

7.7.5 Publish

There is a new PUBLISH control card. With the exception of the OFFLINE and PUBLISH SPECIAL, publish-oriented control cards will be processed and will produce the same effect as in NIPS 1410 FFS. The new PUBLISH card is free format, keyword. Blanks or commas are treated as word delimiters. Any punch in column 72 indicates a continuation. The keywords and their definitions are as follows.

Keyword Examples	Definition
RITID=RITNAME	Identifies name of RIT stored on Permanent Library.
SPECIAL=RITNAME	Identifies name of RIT stored on Temporary Library (replaces 1410 NIPS publish special).
ANSID=NN/RRR	Identifies a set of answers to be published. NN is the query number and RRR is the secondary (RIT) ID used in RASP
COVERPAGE=CDW	Same as the COVERPAGE operand in the old format PUBLISH card.
PAGENO=NN	Same as the PAGENO card.
BODYLINES=NN	Same as the BODYLINES card.
COPIES=NN	Same as the COPIES card.

OUTPUT PROCESSOR (OP)

CLASS=UNCLASSIFIED	Same as the CLASS card. Note that if there are embedded blanks, the classification must be enclosed in quote (') signs.
PARAM='PARAM EXAMPLE'	Parameter allowing the user to introduce up to 60 bytes of information to the RIT at RIT execution time.
DEBUG=NN	Allows the user to designate the number of data blocks to be passed against the RIT. When N is exhausted, an end of file or end of answer set is simulated.
DATE=DDMMYY	Allows the user to specify an 11-character date at run time.

Example:

```
                                                          72
    PUBLISH ANSID=1 RITID=RITEXMP OCVERPAGE=CDW            X
    PAGENO=101 BODYLINES=40 COPIES=2                       X
    CLASS='VERY UNCLASSIFIED'                              X
    PARAM='THIS IS AN EXAMPLE OF THE NEW PUBLISH CARD'     X
    DEBUG=10
```

7.8 Functional Changes

OFFLINE - This will not be treated as an OP function, but rather as a function of the S/360 OS and the print punch utility program.

RECORD - This form of output will be set up as a standard S/360 OS variable length blocked sequential tape file. It will be set up so that at execution time the user can override data set name, retention, output device, blocked to unblocked, and variable to fixed. The file will be closed

after completion of the RIT (NIPS 1410 FFS does not close until completion of the job).

PSCTL - A count of subsets loaded into the processing blocks will be generated. A RIT which references the PSCTL will reference this count. Note that only subsets used by the RIT are loaded; i.e., if the RIT uses only flagged subsets, unflagged subsets will not be loaded. Thus, the NIPS 360 FFS PSCTL might differ from the NIPS 1410 FFS PSCTL.

PSSQ - The NIPS 360 FFS PSSQ differs from the NIPS 1410 FFS PSSQ in both size and content. NIPS 360 FFS OP will process this field; however, it performs no function to provide compatibility or to inform the user of any possible incompatibility.

RIT Executions - NIPS 360 FFS OP differs in the execution of RITs in the following functions:

An answer set may be executed many times by many RITs within the same job.

Standard RITs can be structured and executed within the same job.

A special RIT can be structured once and executed many times within the same job.

Multiple publications of Source Direct can be executed within the same job.

RIT Deck Order - NIPS 360 FFS OP will check for proper statement order as described in this manual. Incorrect order will result in the RIT not being structured.

RUN Deck Order - The order of the run deck will be changed such that all RIT specification decks to be structured will appear <u>first</u>, followed by the SOURCE card, followed by the OP control cards. Note the only change is the placement of the RIT special decks and the ability to do structures and executions in the same executions of OP. Defaults are the same as NIPS 1410 FFS; e.g., an execution of OP with no input specification would result in the assumptions that source is retrieval and mode is <u>ALL</u>.

OUTPUT PROCESSOR (OP)

Publication Order-Source Retrieval - Answers will be published in the order of the answer file (QRT) which may or may not be in the same order as the PUBLISH control cards.

Coordinates - Automatic conversion of coordinate fields based on output length will be performed by OP unless defined as a group or intermediate processing (subroutine, edit) is performed. Also, the use of coordinate fields with the COMPUTE, ARITHMETIC, COUNT, or CONDITION statement will be illegal.

Classification - Failure to provide a classification or designating a classification which is not the same as that designated for the file will result in a warning message to the operator and a warning page preceding the output.

Logic - Specification of BOOL as the last parameter on the CREATE card results in the grouping of AND terms and the separation by OR terms. The Boolean logic interpretation thus becomes consistent with the retrieval language.

DISTRIBUTION

CCTC CODES	COPIES
C124 (Reference and Record)	3
C124 (Record Copy) Stock	6
C240	20
C315	1
C341 (Maintenance Contractor)	10
C341 (Stock)	70

EXTERNAL

Director of Administrative Services, Office of the Joint Chiefs of Staff Attn: Chief, Personnel Division, Room 1A724, The Pentagon Washington, D.C. 20301	1
Director for Personnel, J-1, Office of the Joint Chiefs of Staff, Attn: Chief, Data Service Office, Room 1B738C, The Pentagon, Washington, D.C. 20301	1
Director for Operations, J-3, Office of the Joint Chiefs of Staff, Attn: P & AD, Room 2B870, The Pentagon, Washington, D.C. 20301	1
Director for Operations, J-3, Office of the Joint Chiefs of Staff, Attn: Deputy Director for Operations (Reconnaisance and Electronic Warfare) Room 2D921, The Pentagon, Washington, D.C. 20301	1
Director for Logistics, J-4, Office of the Joint Chiefs of Staff, Room 2E828, The Pentagon, Washington, D.C. 20301	1
Chief, Studies Analysis and Gaming Agency, Attn: Chief, Force Analysis Branch, Room 1D928A, The Pentagon, Washington, D.C. 20301	1
Automatic Data Processing, Liaison Office National Military Command Center, Room 2D901A, The Pentagon, Washington, D.C. 20301	1

EXTERNAL	COPIES
Automatic Data Processing Division Supreme Headquarters Allied Powers, Europe Attn: SA & P Branch, APO New York 09055------	1
Director, Defense Communications Agency, Office Of MEECN System Engineering, Attn: Code 960T, Washington, D.C. 20301----------	1
Director, Defense Communications Engineering Center, Hybrid Simulation Facility, 1860 Wiehl Avenue, Reston, VA 22070----------	1
Director, Defense Intelligence Agency Attn: DS - 5C2 Washington, D.C. 20301----------	5
Commander-in-Chief, Pacific, Attn: J6331, FPO San Francisco, 96610----------	1
Commander-in-Chief, US Army Europe and Seventh Army ATTN: OPS APO New York 09403---	1
Commanding General, US Army Forces Command, Attn: Data Support Division, Building 206, Fort McPherson, GA 30303----------	1
Commander, Fleet Intelligence Center, Europe, Box 18, Naval Air Station, Jacksonville, Florida 32212----------	1
Commanding Officer, Naval Air Engineering Center, Ground Support Equipment Department, SE 314, Building 76-1, Philadelphia, PA 19112	1
Commanding Officer, Naval Security Group Command, 3801 Nebraska Avenue, N.W. Attn: GP22, Washington, D.C. 20390----------	1
Commanding Officer, Navy Ships Parts Control Center, Attn: Code 712, Mechanicsburg, PA 17055	1
Headquarters, US Marine Corps, Attn: System Design and Programming Section (MC-JSMD-7) Washington, D.C. 20380----------	1

EXTERNAL	COPIES
Commanding Officer, US Army Forces Command Intelligence Center, Attn: AFIC-PD, Fort Bragg, NC 28307----------	1
Commander, US Army Foreign Science and Technology Center, Attn: AMXSJ-CS, 220 Seventh Street NE, Charlottsville, VA 22212--	1
Commanding Officer, US Army Security Agency, Command Data Systems Activity (CDSA) Arlington Hall Station, Arlington, VA 22212----------	1
Commanding Officer, US Army Security Agency Field Station - Augsburg, Attn: IAEADP, APO New York 09458----------	1
Commander, Fleet Intelligence Center, Atlantic, Attn: DPS, Norfolk, VA 23511----------	1
Commander, Fleet Intelligence Center, Pacific, Box 500, Pearl Harbor, HI 96860----------	1
Air Force Operations Center, Attn: Systems Division (XOOCSC) Washington, D.C. 20301-----	1
Commander, Armed Forces Air Intelligence Training Center, TTMNIM (360 FFS), Lowry AFB, Co 80230----------	1
Commander, Air Force Data Services Center, Attn: Director of System Support, Washington, D.C. 20330----------	1
Commander-in-Chief, US Air Forces in Europe, Attn: ACDI APO New York 09332----------	1
Commander, USAF Tactical Air Command, Langley AFB, VA 23665----------	1
Commander, Space and Missile Test Center, Attn: (ROCA) Building 7000, Vandenberg, AFB, CA 93437----------	1

EXTERNAL	COPIES
Naval Air Systems Command, Naval Air Station, Code 13999, Jacksonville, Florida 32212------	1
Commanding General, US Army Computer Systems Command, Attn: Support Operations Directorate, Fort Belvoir, VA----------	1
Defense Documentation Center, Cameron Station, Alexandria, VA 22314----------	12
TOTAL	159

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: CSM UM 15-78 Volume V 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): NMCS Information Processing System 360 Formatted File System (NIPS 360 FFS) - Users Manual Vol V - Output Processor (OP)	5. TYPE OF REPORT & PERIOD COVERED 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s)	8. CONTRACT OR GRANT NUMBER(s): DCA 100-77-C-0065
9. PERFORMING ORGANIZATION NAME AND ADDRESS: International Business Machines, Corp. Rosslyn, Virginia	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: National Military Command System Support Center The Pentagon, Washington, D.C. 20301	12. REPORT DATE: 1 September 1978 13. NUMBER OF PAGES: 156
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Copies of this document may be obtained from the Defense Documentation Center, Cameron Station, Alexandria, Virginia 22314.
This document has been approved for public release and sale; its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This volume describes the Output Processor (OP) component of the NIPS 360 FFS. It presents the Output Processor's capability, the type of output produced, and the methods employed in producing this output. It describes the control cards required, a quick reference summary of all cards, and a summary of differences between 1410 NIPS and NIPS 360 FFS concepts of the Output Processor.

This document supersedes CSM UM 15-74, Volume V.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)